NAVAL POSTGRADUATE SCHOOL

MONTEREY, CALIFORNIA

THESIS

AN ANALYSIS OF THE INTEGRATED MECHANICAL DIAGNOSTICS HEALTH AND USAGE MANAGEMENT SYSTEM ON ROTOR TRACK AND BALANCE

by

Mark S. Revor

June 2004

Thesis Advisor:	Lyn R. Whitaker
Co-Advisor:	Arnold H. Buss
Second Reader:	Samuel E. Buttrey

THIS PAGE INTENTIONALLY LEFT BLANK

REPORT DOCUMENTATION PAGE			Form Approved OMB No. 0704-0188
Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instruction, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188) Washington DC 20503.			
1. AGENCY USE ONLY *(Leave blank)*	**2. REPORT DATE** June 2004	**3. REPORT TYPE AND DATES COVERED** Master's Thesis	
4. TITLE AND SUBTITLE: An Analysis of the Integrated Mechanical Diagnostics Health and Usage Management System on Rotor Track and Balance			**5. FUNDING NUMBERS**
6. AUTHOR(S) Mark S. Revor			
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey, CA 93943-5000			**8. PERFORMING ORGANIZATION REPORT NUMBER**
9. SPONSORING /MONITORING AGENCY NAME(S) AND ADDRESS(ES) Goodrich Fuel and Utility Systems			**10. SPONSORING/MONITORING AGENCY REPORT NUMBER**
11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.			
12a. DISTRIBUTION / AVAILABILITY STATEMENT Approved for public release; distribution is unlimited.			**12b. DISTRIBUTION CODE**

13. ABSTRACT (maximum 200 words)

This thesis is concerned with the operational benefit of the Integrated Mechanical Diagnostics Health and Usage Management Systems (IMD HUMS) rotor track and balance (RTB) functionality. The questions addressed are whether there is a savings in flight hours expended on functional check flights (FCF's) when compared to present practices, if there will there be a reduction in directed maintenance man-hours (DMMH) spent on maintenance related to the rotor system, and the impact on Operational Availability. Experiments were conducted using a discrete event simulation model of squadron flight operations and organizational level maintenance. The simulation is generic and can be used in the analysis of other helicopters. Input parameters governing the distributions of maintenance action inter-arrival times were estimated from Naval Aviation Logistics Data Analysis (NALDA) databases and squadron experiences on such systems. The analysis suggests that flight hours spent in FCF are dependent upon vibration growth rate, an unknown quantity, and the maintenance policy for rotor smoothing. Directed maintenance man-hours decrease with increasing numbers of IMD HUMS configured aircraft and further gains are achieved with a maintenance policy suited to a continuous monitoring system.

14. SUBJECT TERMS: Heath and Usage Management Systems (HUMS), discrete event simulation (DES), rotor track and balance (RTB).			**15. NUMBER OF PAGES** 102
			16. PRICE CODE
17. SECURITY CLASSIFICATION OF REPORT Unclassified	**18. SECURITY CLASSIFICATION OF THIS PAGE** Unclassified	**19. SECURITY CLASSIFICATION OF ABSTRACT** Unclassified	**20. LIMITATION OF ABSTRACT** UL

NSN 7540-01-280-5500 Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. 239-18

THIS PAGE INTENTIONALLY LEFT BLANK

Approved for public release; distribution is unlimited

AN ANALYSIS OF THE INTEGRATED MECHANICAL DIAGNOSTICS HEALTH AND USAGE MANAGEMENT SYSTEM ON ROTOR TRACK AND BALANCE

Mark S. Revor
Captain, United States Marine Corps
B.S., U.S. Naval Academy, 1994

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN OPERATIONS RESEARCH

from the

NAVAL POSTGRADUATE SCHOOL
June 2004

Author: Mark S. Revor

Approved by: Lyn R. Whitaker
Thesis Advisor

Arnold H. Buss
Co-Advisor

Samuel E. Buttrey
Second Reader

James N. Eagle
Chairman, Department of Operations Research

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

This thesis is concerned with the operational benefit of the Integrated Mechanical Diagnostics Health and Usage Management Systems (IMD HUMS) rotor track and balance (RTB) functionality. The questions addressed are whether there is a savings in flight hours expended on functional check flights (FCF's) when compared to present practices, if there will there be a reduction in directed maintenance man-hours (DMMH) spent on maintenance related to the rotor system, and the impact on Operational Availability. Experiments were conducted using a discrete event simulation model of squadron flight operations and organizational level maintenance. The simulation is generic and can be used in the analysis of other helicopters. Input parameters governing the distributions of maintenance action inter-arrival times were estimated from Naval Aviation Logistics Data Analysis (NALDA) databases and squadron experiences on such systems. The analysis suggests that flight hours spent in FCF are dependent upon vibration growth rate, an unknown quantity, and the maintenance policy for rotor smoothing. Directed maintenance man-hours decrease with increasing numbers of IMD HUMS configured aircraft and further gains are achieved with a maintenance policy suited to a continuous monitoring system.

THIS PAGE INTENTIONALLY LEFT BLANK

THESIS DISCLAIMER

The reader is cautioned that computer programs developed in this research may not have been exercised for all cases of interest. While every effort has been made, within the time available, to ensure that the programs and data herein are free of computational, logic, and collection errors, they cannot be considered validated. Any application of these programs or data without additional verification is at the risk of the user.

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

LIST OF FIGURES

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF TABLES

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF ACRONYMS

COSSI	Commercial Operations and Support Savings Initiative
DECKPLATE	Decision Knowledge Programming for Logistics Analysis and Technical Evaluation
DES	Discrete Event Simulation
DMMH	Directed Maintenance Man-hour
EMT	Elapsed Maintenance Time
FCF	Functional Check Flight
HUMS	Health and Usage Monitoring Systems
IFP	Inter-Flight Period
IMD HUMS	Integrated Mechanical Diagnostics Health and Usage Management System
IMDS	Integrated Mechanical Diagnostics System
MAF	Maintenance Action Form
MOE	Measure of Effectiveness
NALDA	Naval Aviation Logistics Data Analysis
NAMP	Naval Aviation Maintenance Program
NAVAIR	Naval Air Systems Command
NMC	Not Mission Capable
RTB	Rotor Track and Balance
SME	Subject Matter Expert
TMR	Total Mission Requirement
VATS	Vibration Analysis Test Set
WUC	Work Unit Code

THIS PAGE INTENTIONALLY LEFT BLANK

ACKNOWLEDGMENTS

This thesis would not have been possible without the great support I received from a host of sources. Thanks to everyone that answered a phone call or responded to an email and to everyone that listened when I was wrestling with an issue. Specifically;

Professor Whitaker, thank you for putting up with my self-imposed timelines. Professor Buss, thank you for all of your help with everything, not just the JAVA. Prof Buttrey, thank you for S-plus work, specifically with handling the large set of data.

Professors Paul and Susan Sanchez and Professor Koyak, thank your for listening to the many issues I had as I moved along in this journey.

I have to thank Eric Bechoeffer for making my thesis tour simply outstanding.

To the people at NAVAIR, specifically, Mark Hollins, Mark Bailer, Rich Tullos and Tim Smith, thank you for all of the data, fact checking and opinions. They were instrumental in completing this study.

Mark Little at MCAS New River was extremely helpful. Subject matter expert only begins to describe him. Thank you for all the things that you helped me with especially the quick and detailed responses to my sometimes-daily calls and emails.

And finally to my family, though lately they have been last on my schedule they are always first in my heart and mind. Beth, thank you for your love, support and understanding during this study. Nathan and Megan, thank you for dragging me away from my thesis to come and play with you.

THIS PAGE INTENTIONALLY LEFT BLANK

EXECUTIVE SUMMARY

This study, sponsored by Goodrich Aerospace, attempts to quantify the impact the Integrated Mechanical Diagnostics Health and Usage Management System's (IMD HUMS) rotor track and balance (RTB) functionality will have on a CH-53E squadron. The measures of effectiveness used to quantify the effect of IMD HUMS are; aircraft availability, defined as the amount of time the aircraft was in an up status divided by the amount of time spent in and up or down status; elapsed maintenance time (EMT) for RTB related maintenance; and the ratio of functional check flight (FCF) hours to total flight time.

The study finds that EMT and flight hour ratio decrease as the number of aircraft configured with IMD HUMS increases. For each aircraft configured the percent of total time spent in FCF is reduced 0.23% for current fleet conditions. When an entire squadron is equipped with IMD HUMS, there exists a savings of approximately 60 flight hours per year for a squadron that flies 2100 hours annually. For a squadron commander this means an extra week of flight time for training or fleet support with no additional operating funds. Similarly, EMT decreases with each installation by approximately 75 hours per year. With an average of two maintainers per maintenance action and squadron labor rates equal to $79.71/hour, the cost savings is approximately $12,000 per installation (FY04$) per year. Availability, while directly affected by the number of IMD HUMS installed, was not significantly impacted.

The study was conducted using discrete event simulation utilizing Simkit software. Flight and maintenance data used for input were collected from the Decision Knowledge Programming for Logistics Analysis and Technical Evaluation (DECKPLATE) interface provided by the Naval Aviation Systems Command (NAVAIR). The predictor variables for the simulation include the number of IMD HUMS configured aircraft, the number of maintenance crews, the number of FCF crews, a maintenance rate multiplier, which simulates effects in more rugged environments, and a policy variable. The policy variable was used to run simulations on two different maintenance policies.

THIS PAGE INTENTIONALLY LEFT BLANK

I. INTRODUCTION

In December 2003 the Integrated Mechanical Diagnostic System (IMDS) began Operational Test and Evaluation on three CH-53E Super Stallion helicopters assigned to the Fleet Replacement Squadron (FRS) HMT-302 at Marine Corps Air Station (MCAS) New River. The purpose of IMDS is to enhance maintenance procedures and to improve helicopter operational readiness and fight safety while reducing maintenance-related costs (Pyryt, 2001). IMDS automatically records information about the operation, condition and usage of the aircraft and major components to include the rotor system. One of the expected benefits of using IMDS is in rotor track and balance (RTB) performance. There is a potential for a reduction in flight hours expended on functional check flights (FCFs) compared to present practices. Additionally, there is the possibility of a reduction in directed maintenance man-hours (DMMH) used to make rotor system adjustments during a RTB cycle.

With the potential for benefits there exists potential for degraded availability. With IMDS a high awareness of aircraft and aircraft subassembly state is available. It is the interpretation of data collected by IMDS that determines the need for maintenance on the aircraft. If maintenance policy is set too rigidly the aircraft may have a longer service life but a price will be paid in maintenance man-hours or aircraft availability. In the case of the rotor system, additional FCF hours and DMMH's maybe required to perform RTB if the procedure is required more often when using IMDS than under present conditions.

This thesis investigates the benefits of employing IMDS in the role of RTB and its impact on operations and maintenance for a CH-53E squadron.

A. IMD HUMS

IMDS began as a Commercial Operations and Support Savings Initiative (COSSI) in July 1997 with an agreement between Naval Air Systems Command (NAVAIR) and BFGoodrich. The project is managed by a joint integrated project team from NAVAIR representing the H-53 Helicopters Program Office (PMA-261) and the Multi-Mission Helicopters Program Office (PMA-299) with PMA-261 as the team lead. The current

version of IMDS is Goodrich Corporation's, Integrated Mechanical Diagnostic Heath and Usage Monitoring System (IMD HUMS); BFGoodrich became Goodrich Corporation in June 2001. In this paper IMDS and IMD HUMS are used interchangeably.

IMD HUMS is designed to support a variety of health- and maintenance-related issues including: engine performance assessment; rotor track and balance; vibration absorber tuning; mechanical diagnostics; exceedance monitoring; and usage monitoring. The purpose of the system is to decrease maintenance man-hours per flight hour, improve aircraft safety, provide aircrews with detailed secondary indication capability, increase availability, increase reliability, enable rapid determination of aircraft status and aid in identifying maintenance and logistic actions (Duke, 2001).

B. OBJECTIVES

The approach taken was to conduct the analysis by Discrete Event Simulation (DES) using Simkit software. DES is a modeling paradigm in which the model's state stays constant except at particular events, which can take place at any time. (Law and Kelton, 2000). Simkit is a DES implementation software package that is written in Java and runs on any operating system with Java 2 ^{TM} installed (Buss, Nov 2001).

The DES model follows the organizational level of maintenance described by the Naval Aviation Maintenance Program (NAMP). The full spectrum of maintenance and flight operations on any Navy/Marine Corps aircraft is a complex cycle and difficult to model. This study focuses on the portion of maintenance that is affected by RTB and, where necessary, considers other maintenance aspects in the aggregate.

The level of detail the model is capable of analyzing is dependent on the number of inputs the model given by the analyst. Since all values used to run the simulation are eligible to become variables, the simulation can support investigation into a number of areas of research. For example, in this study the daily flight hours were fixed and the total number of aircraft remain at twelve. Both values could become variables with only a minor modification. Additionally, the CH-53E is the only helicopter simulated. Multiple helicopter types could be included with minimal effort. For this investigation the measures of effectiveness (MOE) are flight hours spent on FCF's and elapsed

maintenance time (EMT) expended on maintenance actions related to RTB. DMMH equals EMT times the number of personnel assigned to that job. EMT is used in place of DMMH to reduce the variation induced in DMMH by the differing number of personnel assigned to RTB maintenance actions. An FCF is defined to be a flight performed to determine if the airframe, power plant, accessories, and items of equipage are functioning per predetermined requirements while subject to the intended operating environment. FCF's are conducted when it is not feasible or possible to determine safe/required functioning by means of ground checks (NAMP, 2001). A RTB is a common procedure conducted during a FCF.

Naval Aviation Data comes in many forms. The data used to drive the simulation came from PMA-261 and through the Decision Knowledge Programming for Logistics Analysis and Technical Evaluation (DECKPLATE) interface, one of the Naval Aviation Logistics Data Analysis (NALDA) systems. In some cases the only information available was a mean value or mean time between events. In those cases an appropriate distribution needs to be assumed and sensitivity to that selection was tested. In the case of raw data, a probability density function was estimated and used in the simulation.

In some areas data simply does not exist. For example, the rate of vibration growth in the rotor head is unknown. Also unknown is the average vibration level of Navy and Marine Corps helicopters since, until now, there has no practical system for continuous monitoring. When data is unavailable but of interest to the research, the gap is filled by using estimates from subject matter experts. The output of the simulation is analyzed to determine what impact the estimates have on the final outcome.

C. ORGANIZATION

Chapter II, Background, consists of important information that amplifies the problem and defines terms need to understand the problems, specifically, what a RTB is and how it is performed. Additionally the section covers some background on maintenance policy and the possible unintended effects of continuous monitoring systems like the IMD HUMS.

Chapter III, Data, covers the sources of the data used to run the simulation and the filtering process that describes how the raw data was transformed into input data. The simulation chapter, Chapter IV, describes the major components of the simulation and initialization process. This section also covers the design of the experiment and the baseline output of the model. Chapter V, Results, is the analysis of the measures of effectiveness. Finally, the Conclusions in Chapter VI find that maintenance man-hours and operational flight time can be recovered from RTB using IMD HUMS. Greater gains will likely be realized if the maintenance policy is adjusted.

II. BACKGROUND

A. HEALTH AND USAGE MONITORING SYSTEMS

On November 6th, 1986 a commercial Boeing Vertol 234 LR Chinook was flying 40 workers out to an oilrig located two and a half miles off of the Shetland Isles when the forward transmission failed. There were only two survivors of the 45 passengers and crew onboard. It was determined that the mishap was the result of the failure of a single transmission gear. The investigation board determined that a monitoring system would have prevented the mishap. Pressure from the oil industry resulted in a more serious development of onboard vibration monitoring systems (Beneoff, 2001). Today's suites of sensors are called Health and Usage Monitoring Systems (HUMS).

Installation of HUMS is primarily to help ensure safety with secondary benefits in improved maintenance. According to an analysis of mishaps by the Charles Stark Draper Laboratory (Davis, 1997), between 1990 and 1996 there were 12 Navy and Marine Corps H-53Es involved in mishaps where the subsequent investigation determined that the accident would have been prevented if a HUMS had been installed. Five of the twelve were class A mishaps, in which there was a loss of life and/or damage in excess of $1,000,000.

In 1991 Stewart-Hughes Ltd produced the first integrated Flight Data Recorder and HUMS certified for use on a helicopter. Though certified, there was no research on where to set the detection alarms for the systems that were monitored. The civil operators servicing the oil industry in the North Sea were the first major users of a HUMS. However, in the early years they experienced very high rates of false alarms resulting in maintenance where a defect was not detected. Though the systems monitoring technology worked the operators were forced to use conservative limits.

The North Sea operators soon discovered that warnings occurred well in advance of a failure, and alarm settings were relaxed. They developed a policy where the first indication didn't warrant a removal unless there were secondary indications (other sources). By 1994 they had enough research to set limits that provided the necessary

degree of safety but were also operationally feasible (Richard Healing, personal communication, 3 March 2004)

In the mid-1990s the U.S. Navy became interested in HUMS technology. In 1997, with the signing of the COSSI agreement, Goodrich Corporation (Fuel and Utility Systems) developed the IMD HUMS. The system has completed developmental test and evaluation and is currently undergoing operational test and evaluation on three CH-53E Super Stallions at HMT-302 and will shortly begin operational evaluation on four SH-60B Seahawks, two at HSL-40 and two at HSL-41.

HUMS, like IMDS, clearly enhances helicopter safety, their other benefits, specifically cost benefits, are unknown. To assess the cost, industry leaders like Sikorsky and aviation agencies like the United Kingdom's Civil Aviation Authority have conducted cost benefit studies (Davis, 1997). The current investigation is not a study of cost benefits; however, the measures of effectiveness used, flight hours expended in functional check flights and EMT used in performance of RTB, contribute directly to operating and support costs.

B. COST/BENEFIT ANALYSES

The Charles Stark Draper Laboratory conducted an initial cost and benefit analysis in 1997 for the IMD HUMS (Davis, 1997). This study assessed the benefits of instituting an IMD HUMS. The overall expectations after the study were that IMD HUMS would reduce inspection time, troubleshooting time, unneeded maintenance actions, unscheduled maintenance, wear damage caused by excessive vibrations, and scheduled maintenance and increase ground and flight safety. It was also concluded that the fielding of IMD HUMS would result in a shift in maintenance practices from scheduled repairs and inspections to a practice based on condition. The study was unable to assign benefits to the expected reduction in aircraft vibrations. While it was acknowledged that the elimination of helicopter vibrations would be of great benefit, no direct cost benefit was assigned because there was no data to indicate the amount of time spent performing a RTB. The study also concluded that money would be saved by

conducting more maintenance at a lower maintenance echelon by more accurately diagnosing a failure to a particular component.

In 2001 The Charles Stark Draper Laboratory conducted a second study that was intended to summarize the results of an IMD HUMS cost and benefit analysis, but was unable to complete its work because of delays in fielding the system (Pyryt, 2001). The report was useful to this investigation as guidance in procedures and sources of data.

Rotorcraft Industries Technology Association (RITA) is a non-profit working consortium of U.S. Industries and universities. Bell, as a member of the RITA HUMS team has developed a Cost Benefit and Analysis Model (CBAM) to provide a common framework for evaluating cost benefits of HUM systems. A common complaint of the software is that it is not customizable (Davis, 1997). The CBAM model was not used in this thesis for this reason and because the focus of this study is on RTB and not directly with cost.

The 1997 Draper study found, among other things, that savings could be expected from the implementation of IMD as a result "reduced-wear damage caused by undue vibrations in the rotor, drive train, and engines of the helicopter."

C. VIBRATIONS

As vibrations in helicopters increase so does the rate at which crew and mechanical parts fatigue. The probability of avionics malfunctions grows and, possibly, greater limitations are placed on the operational envelope. Failure rates for components in fixed-wing aircraft are lower than the rates for similar components installed in rotary-wing aircraft. The impact of vibrations on overall aircraft health was demonstrated in a study conducted by Sikorsky for the U.S. Army Air Mobility Research and Development Laboratory (Veca, 1973). In the study a squadron of H-3 helicopters was configured with rotor-mounted bifilar vibration absorbers and compared to a similar squadron that did not have the device. The two squadrons were similar in size and mission and flew approximately the same number of flight hours over the period of the study, though in different geographic and climatic regions. The results were dramatic. The overall helicopter failure rate was reduced by 48% and corrective maintenance requirements

were reduced by 38.5%. The average reduction in vibration level was 54.3%. Additionally, the aircraft showed a 10% life-cycle cost reduction.

The rotor system is the primary source of vibrations in the helicopter fuselage. These vibrations are expected to filter through to the airframe at the *n* per revolution and 2*n* per revolution frequencies, where *n* is equal to the number of main rotor blades. Additionally, any imbalance of the rotor system such as that caused by out-of-track blades or blade non-uniformities, either natural or from operational use, may result in vibrations at the 1/rev to (*n*-1)/rev frequencies (Wood, 2003).

Vibrations in the airframe are reduced by several techniques. SH-60B uses a spring-mass absorber system. As the name suggests, it is a weight on a spring that is tuned to the vibration frequency. The disadvantage to this system is that the absorption only works for a specific rotational speed. In most cases this limitation is acceptable, and the absorber is tuned to the operational rotation speed of the main rotors, or 100% N_R. However, during an autorotation where rotor speed decays absorption is no longer in effect. A pendulum absorber overcomes the problem of frequency changes by replacing the mechanical mass-spring with a centrifugal spring that is tuned correctly at all operating speeds. These absorbers are designed to reduce vibrations at the *n*/rev or higher harmonics. To reduce vibrations in the 1/rev to (*n*-1)/rev frequencies a RTB is necessary.

D. ROTOR TRACK AND BALANCE

Rotor track and balance is the procedure of minimizing the vibration transferred to the airframe from the main rotor system. Specifically, balancing the rotor is the procedure of reducing the loads from blade non-uniformities that are transferred from the rotor to the hub. Tracking the rotor is the procedure of adjustments to get the tips of the blades to fly in the same plane (Rosen, 1997).

Helicopter pioneers knew of the need to dampen vibrations but believed that to minimize the load the blades needed to be tracked rather than balanced. In-flight methods were not yet available to track the blades. One solution was to use colored chalk on the tips of the blade and then hold a flag at the edge of the rotor arc while the blades

were spinning as the helicopter was parked on the ground. Later optical devices were developed to track the blades in-flight.

In-flight tracking is important since there does not always exist a linear relationship between the flight track and balance and ground track and balance (Mark L. Hollins, personal communication, 19 Feb 2004). Over time it was determined that a rotor system may actually have higher vibrations in a condition of perfect tracking than it had with a somewhat less perfect tip path plane coincidence. This discovery led to methods that collected track and vibration data. The adjustment algorithm then minimized vibrations while keeping the tip path plane variation within defined bounds.

Modern methods involve the use of an optical tracker and an accelerometer to measure the magnitude and velocity of vibrations. Algorithms have been developed to determine what adjustments made to the rotor system translate into reduced vibrations in the cockpit (Johnson, 2004).

E. ROTOR TRACK AND BALANCE WITH IMD HUMS

Presently, the Vibrations Analysis Test Set (VATS) records track data and vibrations in the cockpit at the 1/rev frequency. IMD HUMS takes the measurements further by recording vibrations at 1/rev through *n*/rev in the cockpit and in the cabin of the helicopter with the option of recording track data by mounting an optical tracker on the aircraft. While the IMD HUMS algorithm does not require track data to find a solution, the incorporation of track data is required for certain maintenance procedures. Current procedures differentiate between rotor tuning and rotor smoothing. The former refers to the process of achieving RTB by making large rotor adjustments, which cause significant changes to rotor balance. The latter refers to the process of achieving RTB by continuously monitoring rotor vibration levels with IMD HUMS and making small adjustments to minimize vibration levels from the rotor system.

IMDS has the capability to track and balance the main rotor head without the use of an optical tracker. The use of the optical tracker brings another component into the reliability equation and requires that additional equipment be installed on the aircraft. Optical tracking is also more affected by environmental conditions of sunlight and

conditions of poor contrast between the blades and the sky, though nighttime tracking is possible. Tracker-less balancing is the normal operation for RTB with IMDS. IMDS makes use of all blade-induced frequencies less than the blade passage frequency, *n*/rev. Tracker-less balancing is accomplished through the use of the higher frequencies, which are driven by flapping motion of the blades. Minimizing the vibrations at the higher frequencies reduces the flapping motion; this reduction has the effect of bringing the blade track closer together.

The manufacturer is confident that tracker-less balancing could become the norm; however, users are skeptical about embracing such a change. Presently, the optical tracker is installed for FCF's that require a RTB. FCF's that require an RTB stem from maintenance involving a change of a flight system component, like, for example, a rotor blade. The optical tracker is not used for smoothing operations.

The U.S. Army is developing procedures for a tracker-less RTB following a flight component change. These procedures are being developed to utilize the capabilities of IMDS but are currently prescribed for use only for an *in extremis* situation where the aircraft must fly but there is no optical tracker available.

F. NAVAL AVIATION VIBRATION ANALYSIS PROGRAM

The Naval Aviation Vibration Analysis Program was officially established in 1981. The goals of the program are to avoid catastrophic failures, provide insight and assistance in troubleshooting vibration discrepancies, plan repair and replacement times for aircraft components and to prevent unnecessary periodic disassembly for inspection. Supporting the program are the Integrated Service Support Teams, which collect data from fleet aircraft and set limits based on that information. In order to support the program's goals, vibration thresholds and procedures for monitoring are documented and published for fleet users.

Thresholds are based on statistical data and are measures of velocity in inches per second. The airframe vibrations are categorized by frequency and position on the aircraft. For each frequency and location pair there are advisory and non-operational levels. The non-operational level is chosen so that 95% of the aircraft sampled had lower

levels of vibration. The advisory level is an interval chosen so that at least 68% but less than 95% of the aircraft sampled had lower levels of vibration. A triaxial accelerometer connected to the VATS is mounted at the appropriate position to record vibration levels. The accelerometer is mounted near the cockpit during RTB to measure lateral and vertical velocities. If a vibration in excess of 0.3 ips is recorded in either the lateral or vertical direction, maintenance is required to be performed. There is no limit on the maximum vertical difference between the planes that any two blades fly in to complete a RTB, however; there is an initial requirement for the difference to be less than 0.75 inches before collecting in-flight data for the first time. This requirement exists to allow fewer adjustments and flights to complete a RTB. VATS is used to reduce 1/rev vibrations through the addition of weight to the blades, pitch change rod adjustment and trim tab adjustments (A1-H53CE-VIB-000, 1 April 2002).

G. ENGINEERING JUDGEMENT AND MAINTENANCE POLICY

The safety benefits of continuous monitoring and secondary in-flight failure indication provided by IMDS are apparent. The impacts on operations and maintenance are a bit unclear and difficult to forecast. Optimists have an expectation of healthier aircraft and increased availability while the pessimistic view foresees increased maintenance. The pessimistic viewpoint stems from two sources. The first is the conflict between the use of continuous monitoring equipment and inspection-based limitations. Inspection-based limits are chosen so that the possibility of a failure between inspections is low. Consequently a safety margin is built in to the established limit. With continuous monitoring the same margin is not necessary. Also, IMD HUMS, in addition to continuous monitoring, is collecting data from more sources than previous inspection methods.

For example, the tail rotor drive shaft (TRDS) is made up of seven sections connected by flexible couplings. Presently vibrations are monitored at a single location. Each component of the TRDS produces sinusoidal vibrations at the same frequency because the TRDS turns at a single speed. However, the vibrations combine and translate through the structure to the accelerometer that is periodically installed for an inspection. Once the vibrations are combined they cannot be separated. To determine whether the

vibrations are in excess of the limits the combined input is converted into amplitude at the TRDS frequency using a Fourier Transformation (Mark L. Hollins, personal communication, 19 Feb 2004). With IMDS, accelerometers are mounted on each bearing all of the time. Instead of a periodic comparison to the sum of the vibrations, HUMS monitors directly and continuously. While the case for using different limits for aircraft equipped with IMD HUMS is easy to make, there is no data available by which to set a new limit.

In the case of RTB, VATS monitors vertical and lateral vibrations in the cockpit. With IMDS cockpit and cabin vibrations are recorded. If the cockpit vibrations are within limits but the cabin is not, what should be done? There are no limits assigned to the cabin because those vibrations have not been a part of the inspection in the past.

Further, there is a question on the limits themselves. The present vibration limit for rotor vibration is 0.3 ips. However, the aircraft can be operated safely with vibrations of 0.4 or more. Pilot performance may not be affected until a much higher level, possibly around 0.6 ips. Gupta and Wood (Gupta and Wood, 1981) conducted a study on vibration and mission proficiency on the AH-64 Apache, when it was still called the Advanced Attack Helicopter. Gupta and Wood used ISO 2631 standards, part of which is a chart that estimates the time until performance is affected by vibration in units of g's and the frequency of the vibration in cycles per second (Hz). The CH-53E has a blade passage frequency of 20.86 Hz and a 1/revolution frequency of 2.98 Hz. Using, as Gupta and Wood did, the ISO proficiency curves for vertical vibration, proficiency is expected to degrade after three hours at 2.98 HZ when vibrations are at 0.1g's or 2.0 ips. It would take five hours for vibrations at the blade passage frequency to produce degraded human performance.

The requirement to reduce vibrations to 0.3 ips in the H-53E is not a matter of safety or performance. It is a realistic goal for minimizing vibrations and vibrations at 0.3 ips have a low probability of causing a failure between inspections. If the limitation remains in place, maintenance must be performed whenever the limitation is exceeded. With continuous monitoring there is a possibility that more maintenance will be required. However, early results have shown that vibration levels after rotor tuning with IMDS are

lower than VATS adjusted rotor systems, which could suggest an increase in time between maintenance events.

Policy change is the simplest way to overcome the unknowns. Presently the Army and the Marine Corps are making changes. The Army units involved with Operational Test and Evaluation have implemented a policy that maximizes the ability of the maintainer to exercise his or her own judgment. Rather than declaring the aircraft not mission capable (NMC) when its limits are reached, the maintenance department is allowed to exercise its judgment. The maintenance department retains the option until a level of 0.8 ips is reached at which point the aircraft is NMC until maintenance is performed.

The Marines at HMT-302 are using a slightly more conservative approach. Under the present policy, no action is required for vibrations at or below 0.4 ips. If vibrations captured in the automatic mode of the IMD HUMS exceed 0.4 then the vibrations will be verified on the next flight. That flight is not required to be a functional check flight. If the vibrations are confirmed to be out of limits then the aircraft is NMC upon the return from its mission and will be scheduled for an FCF.

It is unknown to what extent changing policies could reduce FCF flights. The rate at which vibrations grow in the rotor system and the causes of the growth are not fully understood. It can be safely assumed that there is some entropy involved and degradation through blade deformity that occurs during flight operations. The rate at which a tracked and balanced rotor system degrades remains to be determined.

The future of continuous monitoring may not involve variations on vibration levels for which maintenance is optional or required. Policies may be driven by anomalies in the recorded vibrations. Additionally, continuous monitoring should be able to do much more than indicate exceedances. The recorded indications should provide the maintainer some troubleshooting insight by correlating flight regime with failure modes.

THIS PAGE INTENTIONALLY LEFT BLANK

III. DATA

The design philosophy of the simulation is to model the need for a functional check flight by the arrival rate of maintenance actions that require a FCF. This chapter describes the process by which aviation maintenance data was used to develop parametric distributions for inter-arrival times of these maintenance actions.

A. SOURCE

1. Origination

Maintenance data are originated in the squadron where the actions take place. After a maintenance action is completed a member of the work crew enters the information into a database at the squadron. These data are periodically sent to larger databases that support various logistics activities. Flight data are originated and migrated in the same way. A complete description can be found in Vol. II of the NAMP (NAMP, 2001). The data used to support this study came from a variety of sources. Initial work was completed with data made available to Draper labs by the ISST at MCAS Cherry Point along with some raw data from HMT-302 and HX-21. When data requirements became clear, through preliminary models and from the advice of subject matter experts, the final data were collected from NAVAIR's Decision Knowledge Programming for Logistics Analysis and Technical Evaluation (DECKPLATE) online database.

Aviation data are often presented in the form of counts in categories such as number of flights and number of maintenance actions. From the counts mean times between events are calculated. DECKPLATE contains raw maintenance data, from which more detailed and descriptive statistics are available. For example, in addition to reported averages available from more condensed data sources, an estimate of variability and the shape of underlying distributions are possible. The statistics of interest quantify inter-arrival times of maintenance actions.

2. Database Query

Flight and maintenance data were downloaded from this database for every CH-53E in the Marine Corps inventory from the period of 1 January 2001 through 31 December 2003. Three years is used because during this period aircraft experience the full spectrum of maintenance, usually including a Depot visit and all phase maintenance.

In August 2000 a flight safety bulletin effectively grounded most the Navy and Marine Corps' fleet of H-53's. Many of the aircraft were still affected by the bulletin in early 2001. This resulted in a study interval closer to two and half years.

The query to the database returned a chronological ordering of maintenance and flight events by aircraft. Flight information includes the departure date, the flight document number, the recorded total mission requirement (TMR) code, flight hours attributed to the code and the flight leg of the same code. The TMR code is a three-digit alphanumeric designator indicating the reason from the flight. The first character represents the flight purpose; the second character represents the general purpose; and the third character represents the specific purpose (e.g., a 2K2 is a squadron funded functional check flight) (OPNAVINST 3710.7S, 2001).

Maintenance information includes the following maintenance action form (MAF) data: date and time received; date and time maintenance started; date and time maintenance was completed; the organization code of the command performing maintenance; the transaction code, the action taken (AT) code; malfunction code; when discovered code; work unit code (WUC); elapsed maintenance time (EMT) and directed maintenance man-hours (DMMH). The transaction codes are two-character numeric codes used to identify the type of data being reported (e.g., 12 is recorded for equipment work, including engines, involving non-repairable components documented as failed parts). The AT code is a one-character alphabetic or numeric code that describes what action has been accomplished on the item identified by the WUC (e.g., 'R' indicates a removal and replace). The malfunction code is a three-character numeric code used to describe the malfunction that caused the maintenance action on the item described by the work unit code (e.g., 127 for "improper adjustment or alignment" and 731 for "battle damage"). When discovered codes identify when the need for maintenance was discovered (e.g., 'C' for in flight and the flight was aborted and 'D for in flight without an abort). The WUC is a numeric or alphanumeric code that identifies the system, subsystem, assembly, or component on which maintenance is performed (e.g., the 15's are for rotary wing components, 15A70 is the entire main rotor head assembly, and 15A7420 is the main rotor blade extender assembly, a component of the main rotor head). EMT is the actual clock time, in hours and tenths, that maintenance was being performed

on a job. Directed maintenance man-hours are man-hours expended by assigned personnel to complete the maintenance action. DMMH equals EMT times the number of personnel assigned to the job (NAVAIR 01-230HM-8, 2001).

B. DATA FILTERING AND INPUT ANALYSIS

1. Filtering and Grouping

The data set from DECKPLATE amounts to over 500,000 records of the described maintenance and flight related data items. From the DECKPLATE query all flight records and maintenance items involving a repair or a remove and/or replace action were extracted. Specifically, this subset of DECKPLATE records contain maintenance records with AT codes of: B, repair; C, repair with WUC; Q, installation; R, remove and replace; S, remove and reinstall; and T, remove and replace for cannibalization. The subset is further filtered to contain Work Unit Codes that require a FCF after maintenance is complete. The times between (or inter-arrival times) of these maintenance actions are calculated by summing the flight time between similar WUC's. Service times in the form of Elapsed Maintenance Time (EMT) are also collected from this set. From this set only WUC's that had 50 or more non-zero inter-arrival times were kept for the model. Inter-arrival times of zero are attributed to multiple actions of the same type. For instance there is only one WUC code for a rotor blade, but there are seven blades. It is not uncommon to have more than one rotor blade repaired at the same time. If, in one maintenance period, two blades from the same aircraft required maintenance there would be two entries in the maintenance records of that aircraft with the same WUC. While both blades would require maintenance, only one FCF would be flown to verify that the aircraft performance was within defined limits. Table 7, in Appendix A, lists the number of inter-arrivals for each WUC and indicates whether a RTB is required.

The occurrences of a particular WUC cannot be modeled as independent of the occurrence of other WUC's. Since each WUC describes some component on a CH-53E, the usage rate and stress of that aircraft is shared, in part, by all the components that make up the helicopter. Despite the inter-relation, there are no strong correlations between the presence or absence of WUC's across inter-flight periods (IFP). The correlation between any two WUC's is less than 0.45, with the exception of one pair whose correlation is

0.64. However, where there is one maintenance action that requires a FCF, there is a high probability that there are other WUC's that also require a FCF in the same IFP. The data suggest that there is a positive dependence among occurrences of WUC's.

A natural way to account for the dependence among WUC's is to group them at the sub-system level (i.e. grouping WUC's with the same first four digits). The input data for the study took the WUC codes from Table 7 and grouped them at the sub-system level. In some of the groups there are some WUC's that require a RTB and others that do not. The simulation handles this by assigning a probability of an RTB to be the empirical proportion of WUC's that require a RTB. It is not sufficiently accurate, during data collection, to stop at the system level because there are many components within a system or subsystem whose maintenance does not require a FCF. For this reason it is necessary to sift down to the specific WUC's and then build back up to account for the relationship among them. In the simulation the sub-systems are assumed to be independent.

Table 1. Work Unit Code Groups Included in the Model. Parameterized by a Weibull Distribution

Group	Number	Mean	Std. dev	α	β	Sub-System
14E1	217	166.22	145.86	1.07	170.53	MAIN ROTOR FLIGHT CONTROLS
14L1A10*	38	216.45	188.18	1.20	230.48	T/R TANDEM SVOCYLINDER ASSEMBLY
14M1	254	131.62	123.95	1.00	131.88	MN RTR SVOCYL CTRG SPR CYLINDER ASSY
15A6	1302	62.01	72.91	0.80	54.93	MAIN ROTOR BLADE
15A70	1545	107.14	113.64	0.89	101.17	MAIN ROTOR HEAD
15AD100*	346	108.87	85.67	0.94	106.10	TAIL ROTOR BLADE ASSEMBLY
15AF400*	90	150.96	140.53	1.00	151.52	HEAD/HUB ASSEMBLY
22600*	1057	56.42	66.57	0.79	49.62	T64 ENGINE
2261	230	131.88	124.95	1.00	131.58	VARIABLE GEOMETRY
22662*	270	114.32	110.91	0.96	112.82	MAIN FUEL CONTROL
26D1	247	121.45	123.48	0.84	112.12	MAIN GEAR BOX
* Single WUC						

2. Inter-arrival Times and Service Times

The observed inter-arrival times from the filtered set of WUC's are pooled as indicated in Table 1. Some WUC are not matched with any others at the sub-system level. The grouped inter-arrival times are used to fit a Weibull distribution using maximum likelihood estimation. The density for the Weibull distribution with shape parameter $\alpha>0$ and scale parameter $\beta>0$ is:

$$f(x) = \frac{\alpha}{\beta^{\alpha}} x^{\alpha-1} e^{-(x/\beta)^{\alpha}}, x \geq 0.$$

Estimates of α and β are given in Table 1 for each WUC group. The distributions of the arrival times are also estimated non-parametrically using the empirical distribution based on the data. In nearly all cases the Weibull distribution fit the data very well in the right tail but not in the left. The empirical distributions have more weight on short inter-arrival times than do the fitted Weibull distributions. Sample runs of the simulation comparing the Weibull distribution and the empirical distribution indicate that the simulation results are not sensitive to the differences: therefore, the Weibull distributions are used to model inter-arrival times.

The service times of the grouped data are handled differently to ensure that aircraft spend an appropriate amount of time in maintenance before conducting a FCF. In this simulation, service times are generated using Weibull distributions with parameter estimates based on all the historical service times of the WUC's that comprise a group. The inter-arrival distributions were generated from historical data excluding those observations with inter-arrival times of zero. The inter-arrival times of zero indicate that a maintenance action was initiated for more than one instance of a particular WUC. Additionally, the inter-arrival times are based on groups of WUC's: therefore, in addition to repeated WUC's in the same IFP, there could be WUC's from the same groups. The probability of multiple maintenance actions from a particular group is estimated from the historical data. The distribution of multiple maintenance actions is estimated non-parametrically from the number of WUC's that occurred in each IFP. Additionally, the distribution is conditioned on the occurrence of at least one WUC from the group. For example, if a WUC from the group 14E1 arrives during an IFP in the simulation a random number from the conditional empirical distribution determines how many

maintenance actions from that group will be serviced during this IFP. Suppose that the result is two; then the distribution of service times is queried twice and the total service time for the group is the sum of those two random draws. Figure 1 is plot of the empirical distribution function $F_n(x)$ of the 14E1 group.

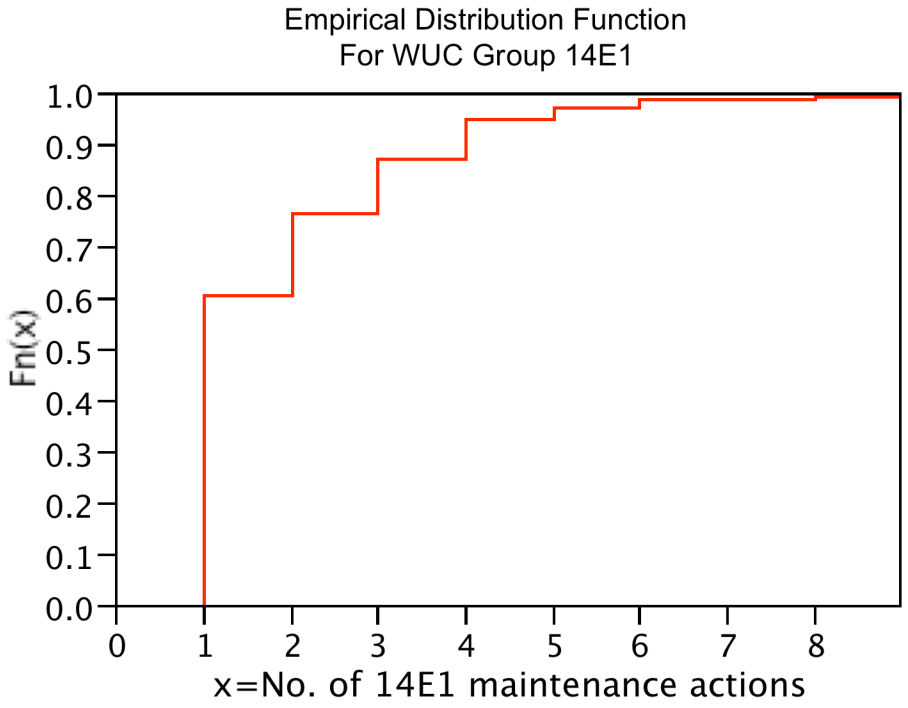

Figure 1. Plot of the empirical distribution function $F_n(x)$ for multiple arrivals of WUC group 14E1

C. GAPS AND ASSUMPTIONS

Despite all the data on hand, there are several requirements that were not available. One of these items is the number of flights required to complete a RTB. Subject Matter Experts (SME) interviewed were able to fill in this gap based on their experience. Responses were in concurrence that at least two adjustments were necessary but differed on the maximum, which ranged from five to seven adjustments. The number of DMMH spent on RTB events is also not available. There is a WUC to log hours spent on adjustments made to complete the event but not for the time spent to install and remove the necessary equipment. For this value SME's were also used.

Additionally, there are no available data to indicate what the vibration levels are after a flight nor the rate of increase in vibrations from flight to flight. These data are important to estimate the time from a RTB event until a rotor tuning is required due to excessive vibrations due to excessive vibrations. In general, a RTB for excessive vibrations is not a common event for the H-53. This may be because of the frequency of a RTB for reasons of component replacement. Because there is not a historical basis for a vibration model, the simulation uses a simple linear model based on a previous study on HUMS and the SH-60 (Schafer and Haas, 2002):

$$\text{vibration} = \text{growth}*\text{FltHrs} + S + \eta(0,0.01),$$

where 'vibration' is the cockpit vibration in ips; 'FltHrs' is the number of flight hours since the last RTB event; 'growth' is the rate at which the vibrations grow; 'S' is the vibration level after the last RTB modeled as a 0.17, 0.29, 0.24 triangular distribution for the VATS/ATABS and parameters 0.12, 0.24, 0.19 for IMD HUMS; $\eta(0, 0.001)$ represents a random Normal noise factor with a mean of zero and variance of 0.001. Schafer and Haas used a value of 0.01 for 'growth' in their model of the SH-60B. The CH-53E has fewer rotor-tuning events for excessive vibrations and so a growth factor of 0.005 is used.

D. MESSY DATA

Aviation data in general can be described as messy data. All of the variables had observations with extreme values. For example, the Automatic Flight Control System's pitch servo cylinder assembly has a mean service time of 3.11 hours. 165 of the 182 observations are between zero and 7 hours of EMT. Of the remaining 17 observations 16 are between 7 and 20 hours and the final observation is 99.1 hours of EMT. There were also cases of apparent lost data. An aircraft on deployment to Djibouti logged 29.5 flight hours over a period of one month with out a single maintenance action. This feat is all the more unlikely since one of the TMR codes indicated that it was conducting a FCF. In the database there were flights and maintenance actions for 17 aircraft with Bureau

Numbers that did not exist. There were also FCF's listed as having been flown for periods greater than the maximum endurance of the aircraft.

Extreme values for maintenance times were kept with the analysis since outliers do occur and are not always data entry mistakes. However, the maintenance and flights logged against the fictitious aircraft were deleted with two exceptions. In both cases the intended Bureau Number was very apparent. FCF's beyond the endurance of the aircraft were assumed to be multiple flights logged as one.

IV. MODEL AND SIMULATION

A. MODEL

The model used for the simulation is based on the organizational level of maintenance as described in the Naval Aviation Maintenance Program (NAMP, 2001). Figure 2 is a simplified diagram that represents basic flow of the model and simulation listener arrangement. A helicopter will conduct flight operations until it meets with a failure event. When the Maintenance component hears the failure event the failing aircraft is passed to Maintenance. After service is completed the aircraft is up or it requires a FCF. If a FCF is required the Functional Check Flight component hears the event and receives the helicopter. If the aircraft does not require a FCF, Flight Operations hears the event and it receives the helicopter. In the Functional Check Flight component the helicopter will either complete its FCF and return to Flight Operations or another failure will occur and it will be returned to the Maintenance component. An aircraft can move directly from Flight Operations to the Functional Check Flight component if it requires a rotor-tuning event due to excessive vibrations.

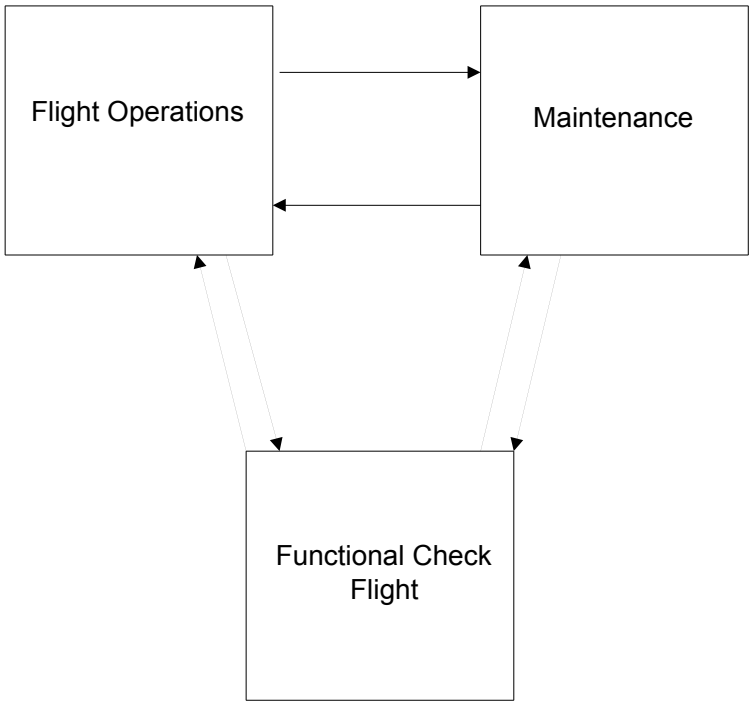

Figure 2. Major model components and their relationship.

1. The Helicopter Object

The simulation object that flows through the model is the helicopter. The helicopter object holds all the parameters for failure modes, failure arrival times, service time, conditional probability of multiple maintenance, vibrations levels, FCF flight times, time to install and remove vibration analysis equipment and the number of RTB iterations until complete. Additionally, a helicopter object can be configured with IMD HUMS. An IMD HUMS configured aircraft will have different parameters for time to configure for a RTB, time to remove gear after a RTB and the number of adjustments required for completion of a RTB.

The helicopter object keeps track of its own flight hours and maintenance requirements. When the simulation is run each helicopter uses its given parameters to determine when each scheduled and unscheduled maintenance event will arrive. As flight hours accumulate the helicopter checks to see if maintenance is required. Flight hours are determined by the flight schedule and recorded against an individual helicopter in the flight operations component.

2. Flight Operations

The flight operations component runs a simulated flight schedule. The flight schedule can be tailored to meet the requirements of the user. For this simulation the baseline case reflects a squadron of 12 aircraft flying 2310 scheduled flight hours annually (hours that do not include FCF's). Squadron operational flight hours and number of available aircraft are tracked for output statistics.

A helicopter object experiences five major events in the flight operations component; up, takeoff, land, down and FCF (See Figure 3). Upon initialization of the simulation, all helicopters are placed in a first-in-first-out (FIFO) queue. Flight schedules arrive at prescribed intervals. Upon the arrival of a schedule, the number of required helicopters for the first event is pulled from the queue. A helicopter can fail at three points during flight operations; pre-flight, in-flight or post-flight. If more than one type of failure is due to occur arrive after a flight they are all entered into a list of needed maintenance. A query is made before take-off to see what failures will occur after the flight hours are logged. For each expected failure there is a 0.5% chance of the aircraft aborting during pre-flight. There is a 1% chance per expected failure mode for an in-

flight abort. These probabilities reflect historic occurrences of in-flight and pre-flight aborts. If the aircraft fails in flight a uniform random number between zero and the total flight length determines the number of flight hours logged against the aircraft. If a helicopter does not fail before, after, or during a flight it is returned to the queue of up aircraft. If it does fail, it enters a FIFO queue of aircraft awaiting maintenance in the maintenance component of the simulation. A third condition occurs when the aircraft has not had a failure, but the aircraft vibrations have increased to a level where a rotor-tuning event is required. Under one maintenance policy, the aircraft is sent to the FCF component. Under a second policy, the vibrations are reduced on the next operational flight. If there are not enough aircraft in the queue to support an event on the flight schedule, a dropped flight is recorded and the simulation proceeds.

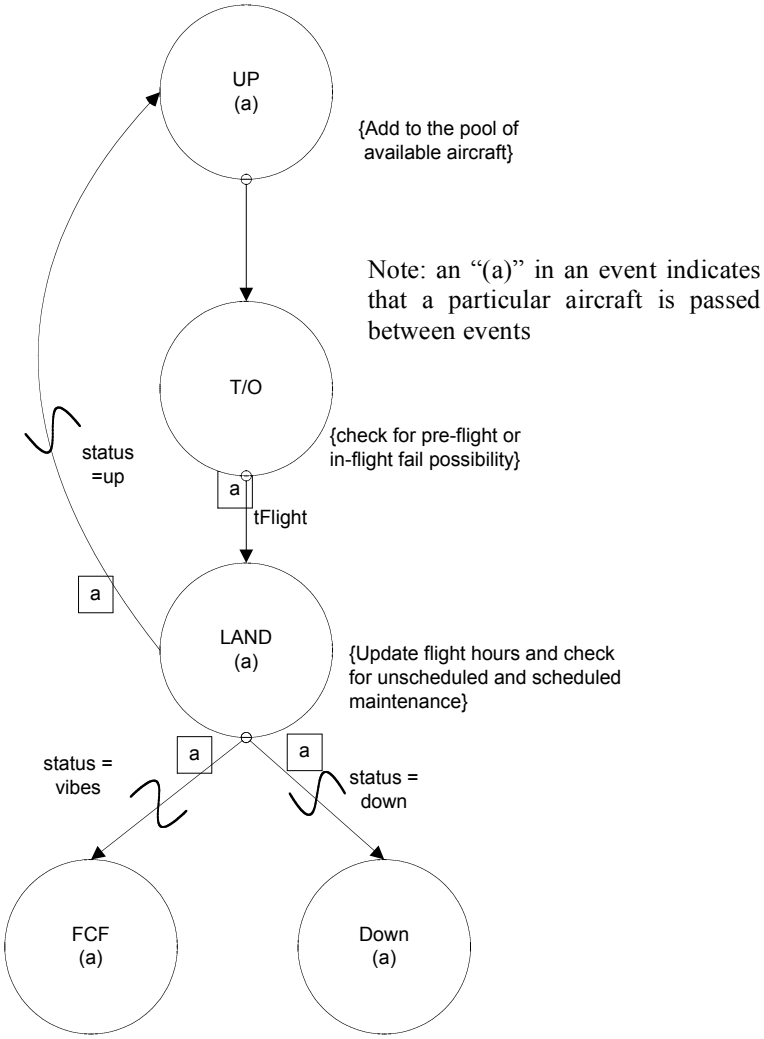

Figure 3. Flight Operations Event Graph

3. Maintenance

This component uses units of 'maintenance crews' to determine the number of actions that can be simultaneously worked on rather than resolving down to the number of individual maintainers. This level of resolution is considered acceptable since the total number of maintenance personnel is less important than the type of qualifications they possess. The number of crews can be varied for the sake of analysis. In this scenario a range from 5 to 10 is used.

The Maintenance cycle begins with the arrival of a scheduled or unscheduled maintenance action. This component receives a downed helicopter, conducts maintenance and either passes it to the functional check flight (FCF) component or returns it to the up queue in flight operations. When a helicopter meets with scheduled or unscheduled maintenance it is added to a FIFO maintenance queue. If a maintenance crew is available work will begin immediately; otherwise the aircraft will remain in the queue until a crew becomes available. Service time is dependent on the type of failure. After the action is complete the time until the next failure is determined and stored with the helicopter object. If it is an unscheduled action then the next occurrence is randomly determined by the distribution of its failures. If the completed maintenance is scheduled then the next inspection time is stored with the helicopter. If the list of required maintenance is not empty then the aircraft re-enters the awaiting maintenance queue. If there are no maintenance actions remaining, the helicopter is either returned to the up queue in flight operations or sent to a FIFO queue of aircraft awaiting a FCF.

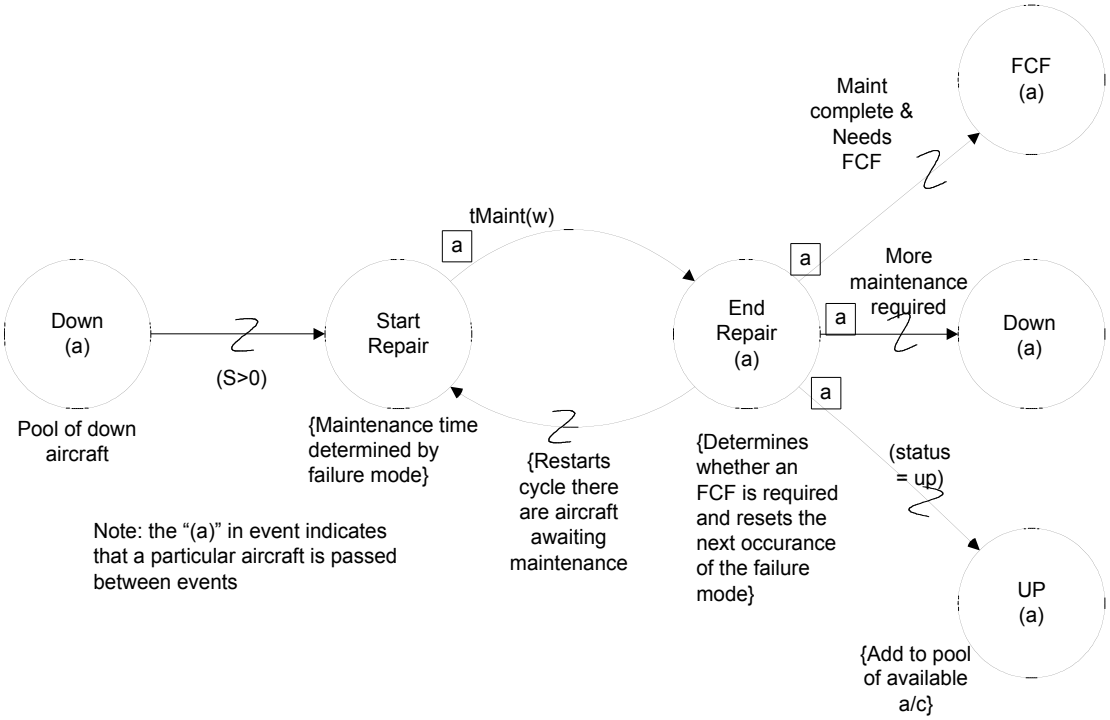

Figure 4. Maintenance Component Event Graph

4. Functional Check Flight

This component consists of ground checks and flight checks (see Figure 5). If the maintenance action does not require a RTB then the ground checks are skipped and only the flight portion of this component are completed. The number of simultaneous functional check flights that can be conducted is dependent on the number of FCF crews available and the number of VATS/ATABS or optical trackers that are ready for issue (RFI). These levels can be varied for the simulation to determine their effect on the time the aircraft is down. In this scenario the number of FCF crews is adjusted from one to four. The number of VATS/ATABS is set to three. The number of optical trackers is set to four.

An aircraft placed in the FCF queue will wait until an FCF crew is available. If the check flight requires the use of VATS/ATABS and that equipment is not available the aircraft is passed over but remains at the top of the queue. The same procedures apply for

a RTB requiring the optical tracker used by IMD HUMS. If a RTB is not required the aircraft conducts only the flight cycle of this component.

If the required gear is available the ground track begins after a delay for gear installation. Before the functional check flight takes place the number of adjustments needed to complete the ground track phase and the number of adjustments required until the aircraft is smoothed in the flight phase is determined. The distribution of required adjustments is held by the helicopter object and is different depending on whether or not the helicopter is configured with IMD HUMS. The simulation repeats the cycle until the required number of iterations has been met. The same possibilities of pre-flight, in-flight, or post-flight failure exist for FCF's. At the end of a flight, if there were no failures, the helicopter will be added to the queue of up aircraft and returned to the flight operations component of the model. If a failure occurred the aircraft is sent to the awaiting maintenance queue.

Table 2. Event Graph Terms

Event Graph	**Term**	**Meaning**
Flight Operations	tFlight	Time of flight
Maintenance	S	Number of Maintenance crews available
Maintenance	tMaint	Time to service the aircraft
FCF	tConfig	Time to configure the aircraft with vibration analysis equipment
FCF	tGndC	Time required to collect ground track data
FCF	tAdj	Time required to make adjustments to the rotors
FCF	tFCF	FCF time
FCF	tDerig	Time required to remove vibrations analysis equipment

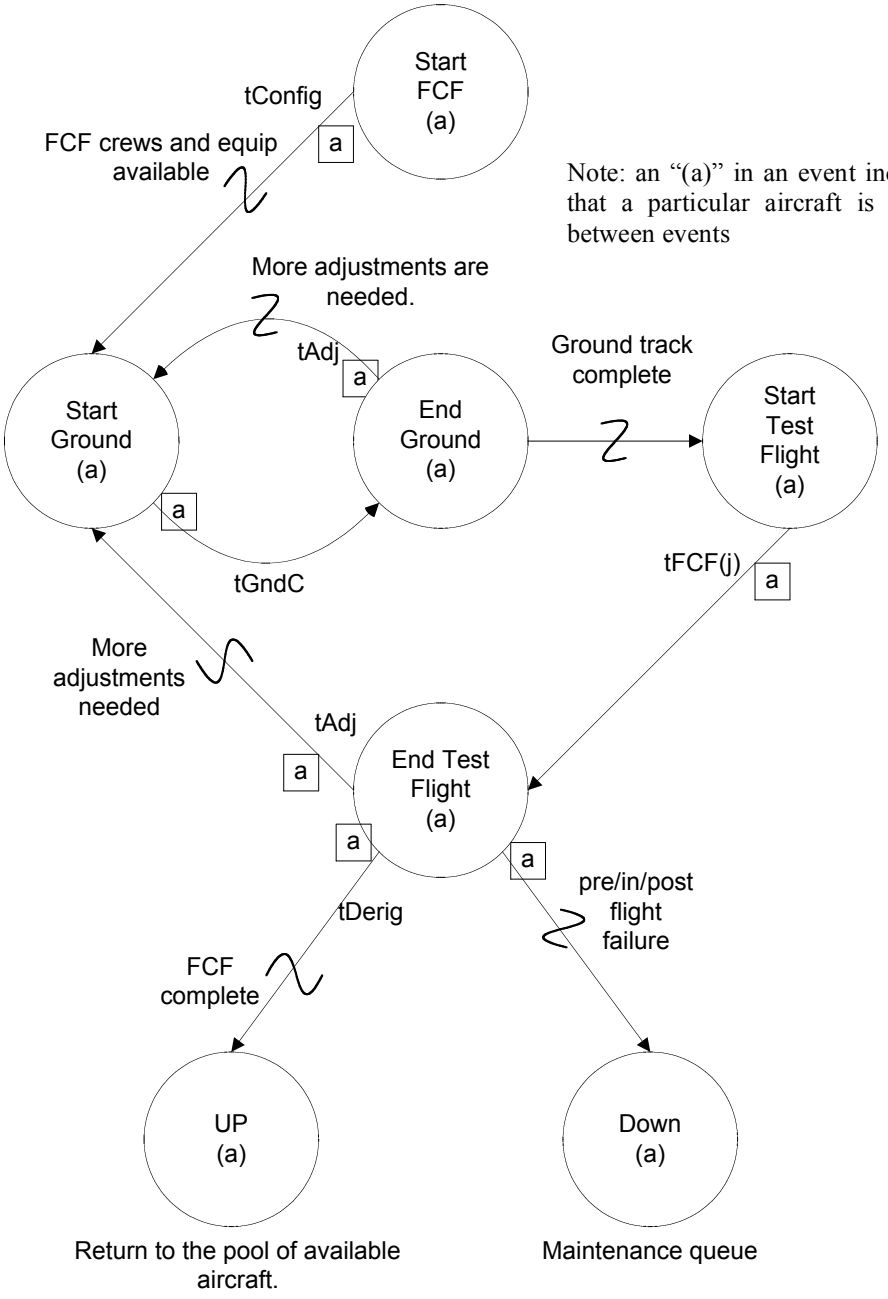

Figure 5. Functional Check Flight Event Graph

5. Initialization

When the simulation is run the user-entered parameters are passed to a squadron generator, a schedule generator and the major simulation components (see Figure 6). The squadron generator creates the requested number of helicopter objects, gives them the characteristic described by the user and enters them into the queue of up aircraft in Flight Operations in a random order. The helicopters are a mix of IMD HUMS configured

aircraft and non-IMD HUMS configured aircraft. The mix depends on user specification. Each instantiated helicopter generates a random time until the first failure for each failure mode. The time until the first scheduled maintenance event (e.g., phase) is also randomly set. After that event takes place the next failure is determined by its scheduled interval. The flight schedule generator creates a list of flight events. Each event has a take-off time, flight duration and the number of aircraft required to fly in that event. At prescribed intervals a schedule is passed to the Flight Operations component where it is executed. The Maintenance and Functional Check Flight components are given the value of the number of maintenance crews and FCF crews, respectively.

Several measures are in place to guard against a bias toward the starting conditions. At the start of the simulation all the helicopters are in an up status, which is an unlikely condition for any squadron to find itself in. The simulation is run for a period of one simulated year before statistics are kept to ensure that the results are not biased. The randomization of the time until the first phase maintenance event significantly aids in reducing the amount of time required to remove the initial condition bias.

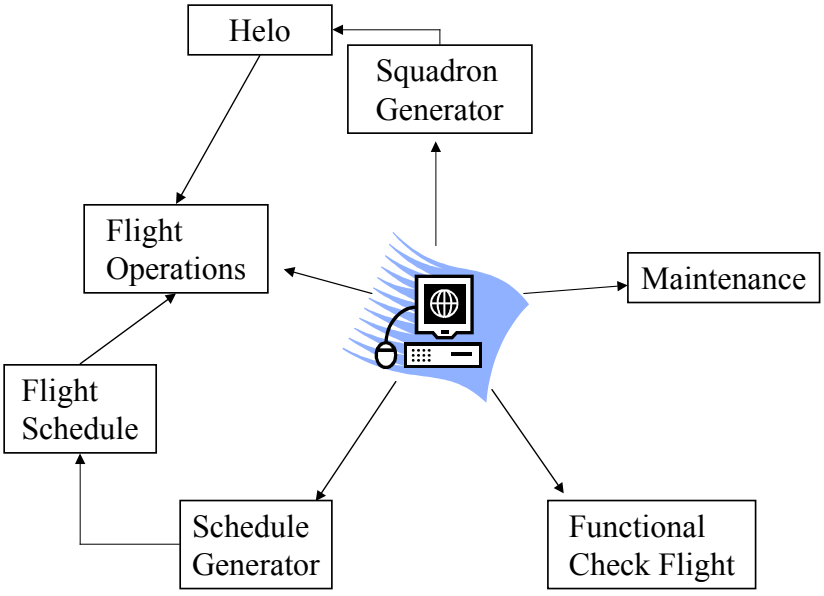

Figure 6. Initialization

B. RANDOM NUMBER STRUCTURE

Throughout the simulation the helicopter object is queried for a random variate. When queried the helicopter returns a random variate defined by a distribution that describes the event (e.g., a time to failure or time to repair). To prevent a correlation in returned random numbers between helicopters, a single random number generator instance is referenced by each distribution.

The mechanics of the pseudo random number generator are described in Figure 7. At the top level is a random number generator that, when requested, returns a number between zero and one. The default congruential random number generator in Simkit uses an unordered sequence of 2^{32} numbers between zero and one. By default a random number object will begin returning numbers from the start of the sequence. The random number object can be given a seed to start elsewhere in the sequence. The number returned is used by the intermediate level to return a number to a helicopter object at the bottom level according to a distribution that describes a particular event.

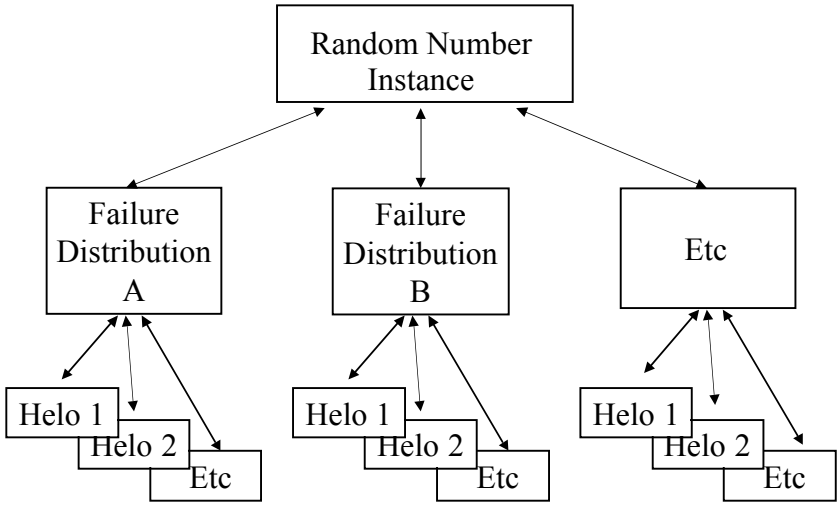

Figure 7. Random Number Tree

C. DESIGN OF THE EXPERIMENT

1. Variables

Five variables are considered as predictors of the percentage of total flight time spent on FCF's, elapsed maintenance time spent on RTB related maintenance and Aircraft Availability. The first variable captures the configuration of the aircraft, that is, the number of squadron aircraft configured with IMD HUMS. Te variable takes integer values between 0 and 12, inclusive. The second variable is the number of maintenance crews, which takes integer values between 5 and 10, inclusive. The third is the number of FCF crews, which takes values between 1 and 4, inclusive. The fourth is a multiplier on the arrival rates to simulate operations in an environment that may be more or less taxing on the aircraft. The maintenance rate variable takes values between .7 and 1.2, inclusive, in increments of 0.1. An example of a rate increase would be desert operations. In this environment the sand causes an increase in rate of rotor blade replacement and affects other components similarly. In the simulation, after an inter-arrival time is generated it is multiplied by this factor before it is placed on the event list. The final factor is a categorical variable for maintenance policy. Two policies will be examined. The first policy will place the aircraft in a down status upon return from a flight if, during the flight, a vibration level greater than 0.3 ips was detected. The second policy would allow for rotor smoothing, the process of making small adjustments to keep the vibrations minimized, without placing the aircraft in a FCF, and therefore not mission capable status.

The simulation is run for the equivalent of 50 years per design point. As the simulation is run, time varying and tallying statistics are recorded. At the end of each year of simulation time the sum of the flight hours, FCF hours and DMMH and the mean of the number of available aircraft are recorded as an observation. The first observation is deleted to guard against initialization bias. After the 50^{th} recorded observation the simulation is reset, new parameters are entered according to the design of the experiment and the cycle repeats. The result is 100 records of the grand means and standard deviations from each MOE. The design of the experiment ensured that the 100 design points had sufficient coverage of the range and interactions of the variables.

2. Latin Hypercube

A full factorial design of all combinations of the five predictor variables would require 4,608 design points. A more efficient method is to use a Latin Hypercube design (Cioppa, 2002). The design used for this experiment uses design points that are nearly orthogonal. The orthogonal Latin Hypercube design, and nearly-orthogonal designs, enable a design that covers all factors and levels sufficiently to detect non-linear relationships without an exceedingly large number of design points. The design of this experiment used 100 design points with the variables and values as shown in Table 3. The input value for each variable of each design point is listed in Appendix A.

Table 3. Orthogonal Latin Hypercube Design Points

Variable	Low Level	High Level
Number of A/C with IMD HUMS	0	12
Number of maintenance crews	5	10
Number of FCF crews	1	4
Maintenance rate	0.7	1.2
Policy (Tuning requires an FCF =1)	0	1

D. VERIFICATION

The simulation was run using the described fleet data and predictor variable set to model normal fleet levels. That is, no IMD HUMS configured aircraft, seven maintenance crews, two FCF crews, no maintenance rate multiplier and the current maintenance policy (an FCF is required if the vibrations increase beyond 0.3 ips.). Marine CH-53E squadrons spent 17.5% of their flight time in FCF's according to historical data available from DECKPLATE. The output of the simulation averages 17.3% with a standard deviation of 1.1%. This is the baseline against which changes will be measured.

THIS PAGE INTENTIONALLY LEFT BLANK

V. RESULTS AND DISCUSSION

To simplify the discussion, the model variables are given names in this section rather than the descriptive labels used in the previous section. The label 'imds' represents the variable for the number of aircraft configured with IMD HUMS. The label 'mcrew' represents the variable for the number of maintenance crews. The label 'fcrew' is for the number of FCF crews assigned. The label 'rate' is for the maintenance rate multiplier variable. The label 'policy' is an indicator variable. A value of one (or true) indicates a requirement to conduct a RTB for recorded vibrations above the threshold. A value of zero (or false) indicates the absence of that requirement. This is a situation where rotor smoothing is authorized during an operational flight instead of during a FCF.

A. ELAPSED MAINTENANCE TIME

The EMT used in the installation and removal of vibration analysis equipment and the time required to make adjustments to the main rotor system based on the algorithm of the RTB software are tallied throughout the simulation. The multiple regression model is the result of using a stepwise algorithm of adding and dropping terms by using the Akaike Information Criterion (Akaike, 1974). The multiple R^2 for this least square fit is 0.9939 and the residual standard error is 31.19 hours. Table 4 gives the estimated coefficients, their standard errors and the corresponding *t*-statistics to test the partial effect of each variable. Diagnostic plots of the residuals support the multiple regression modeling assumptions of Normal errors and homoscedasticity. The variable 'rate' has the heaviest weight on the expected value of EMT, followed by 'imds' and the interaction between 'imds' and 'rate'. The weight of the coefficient on 'rate' is not a surprising result. As the variable 'rate' decreases the time between maintenance actions also decreases. The fitted model increases the amount of maintenance time when repairs occur more often.

The interaction between 'imds' and 'rate' suggest that as rate decreases the benefits of having more aircraft configured with IMD HUMS increases. For example, with all other variables held constant, the effect of configuring one, or one additional,

aircraft with IMD HUMS when the 'rate' variable equals one is a reduction of EMT by 75.87 hours per year. If the 'rate' variable were 0.7 instead of 1.0 then the addition of one IMD HUMS is a reduction in EMT by 105.45 hours per year. Typically two maintainers accomplish this maintenance action. With squadron labor rates reported at $79.71/hours the savings equates to approximately $12,000 (FY04$) per installation per year. The interaction of 'imds' and 'rate' is illustrated in Figure 8. In Figure 8, the variables 'mcrew', 'fcrew' and 'policy' are held constant at 7, 2, and 1, respectively. The slope of the line when 'rate' is equal to 0.7 is steeper than the line when 'rate' equals one. Note that though the slope of the estimated EMT when 'rate' equals 0.7 has a greater absolute value, the expected EMT for a 'rate' value of 1.0 is always smaller.

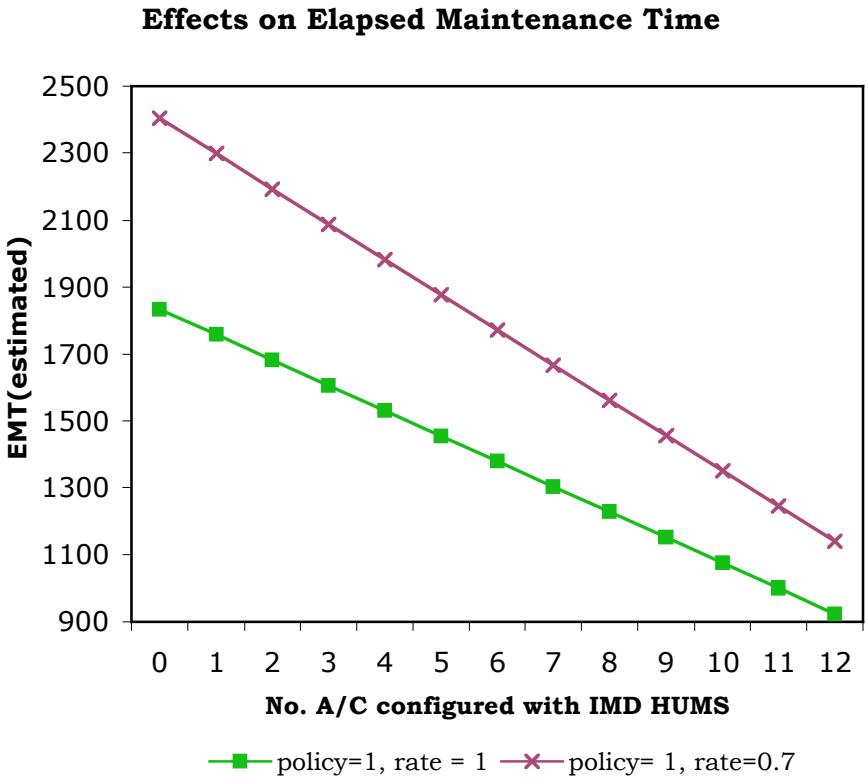

Figure 8. The effects of 'rate' and 'imds' on EMT (estimated)

An interaction between 'imds' and 'policy' was anticipated; however, the stepwise process did not select the interaction. The expectation exists because of the change to continuous monitoring of vibrations that will come with the IMD HUMS. With continuous monitoring there would be near perfect knowledge of-rotor induced vibration.

Presently, with non-IMD HUMS aircraft, a pilot would need to return from a flight and report high vibrations on a post flight report before vibration would be addressed. The pilot is not likely to detect vibrations at the 0.3 ips threshold. Therefore, IMD HUMS aircraft will incur DMMH that would not normally be performed because the monitoring will detect vibrations greater that 0.3 ips anytime they occur.

Table 4. EMT Regression Details

Coefficients	**Estimate**	**Std. Error**	**t-value**	**Pr(>|t|)**
(Intercept)	3910.315	112.838	34.654	<2e-16
Imds	-174.457	6.127	-28.475	<2e-16
Mcrew	-24.318	15.692	-1.550	0.1247
Fcrew	-32.780	16.430	-1.995	0.0490
Rate	-2039.484	109.880	-18.561	<2e-16
Policy	62.119	34.026	1.826	0.0712
Imds:rate	98.589	6.188	15.931	<2e-16
Mcrew:policy	-7.541	4.476	-1.685	0.0955
Mcrew:fcrew	3.611	2.226	1.622	0.1083
Mcrew:rate	19.822	14.446	1.372	0.1734
F-statistic: 1.894e+04 on 2 and 97 DF, p-value: < 2.2e-16				

The vibration rate used in the simulation is 0.005 ips increase per flight hour, plus a noise factor. If the true rate of vibration increase were greater than 0.005 ips the number of RTB's stemming from high vibrations would be likely to increase. Conversely, if the rate were less then the expectation would be for fewer RTB events. If rotor smoothing, the process of making small adjustments periodically to minimize rotor vibrations, were allowed then the additional RTB events would not take place. To demonstrate the effect of vibration growth rate on EMT, the simulation was run again, this time holding the variables 'mcrew', 'fcrew', and 'rate' at 7, 2, and 1.0, respectively, and varying the value of 'imds' and 'policy' in a full factorial design.

Figure 9 illustrates that although the difference is not extreme, a higher vibration growth rate will reduce the benefits of IMD HUMS on EMT when rotor smoothing requires a full RTB. The standard errors for the expected EMT spent on RTB decreased from 140 to 80 as the expected value decreased.

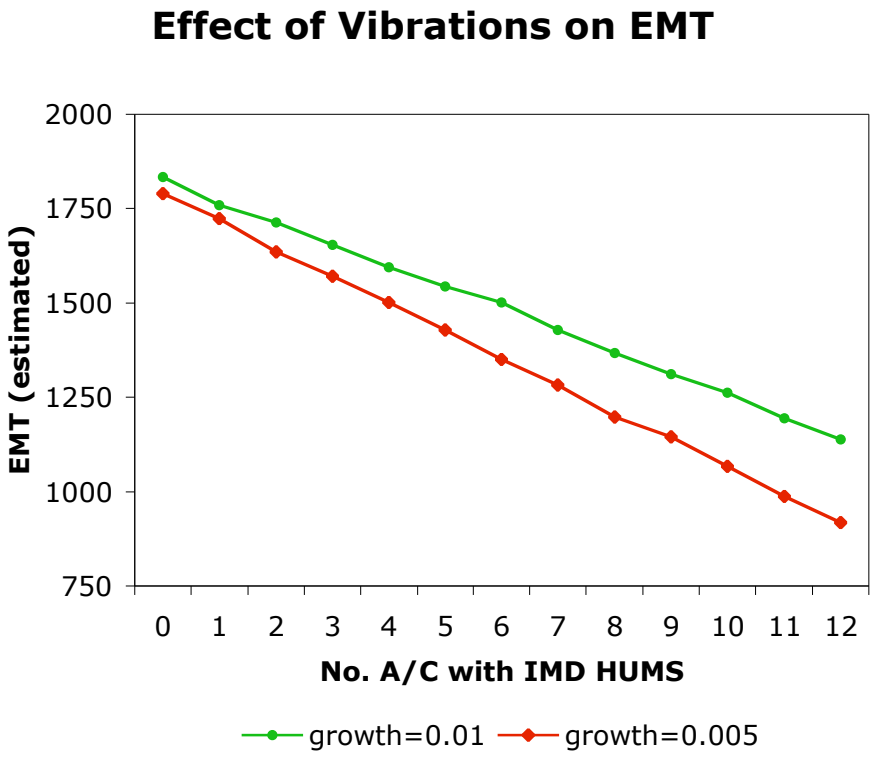

Figure 9. Effects of Vibration Growth Rate on expected EMT.

B. AVAILABILITY

Availability, A_O, is a time-varying statistic and defined as the amount of time spent in an up status divided by the sum of up status time and down status time.

$$A_O = \frac{\text{time in up status}}{\text{time in up status} + \text{time in down status}}$$

For this simulation, since there is no out of service time (e.g., Standard Depot Level Maintenance), the total up and down status time is equal to the total run time.

The change in availability over the range of the parameters was moderate. The maximum value was 11.1 and the minimum value was 9.3. While A_O is not greatly affected by the effect of variable value change, the variable with the most influence is 'rate'. If the simulation is run with 'rate' held constant at 1.0, then the variable representing the number of aircraft configured with IMD HUMS is the most significant. However, A_O is reduced to a range of 10.995 to 10.696.

Table 5. Availability multiple regression details

Coefficients:	**Estimate**	**Std. Error**	**t value**	**Pr(>|t|)**
(Intercept)	7.013	0.394	17.778	< 2e-16
Imds	0.059	0.019	3.036	0.003
Mcrew	0.316	0.052	6.082	0.000
Fcrew	0.171	0.055	3.121	0.002
Rate	3.050	0.393	7.752	0.000
Policy	-0.719	0.170	-4.234	0.000
mcrew:rate	-0.235	0.048	-4.869	0.000
mcrew:fcrew	-0.018	0.008	-2.357	0.021
Fcrew:policy	0.065	0.022	3.024	0.003
mcrew:policy	0.036	0.015	2.421	0.018
Imds:mcrew	-0.004	0.002	-1.601	0.113
Imds:fcrew	-0.008	0.003	-2.256	0.027
rate:policy	0.259	0.154	1.684	0.096
F-statistic: 63.54 on 12 and 87 DF, p-value: < 2.2e-16				

The model is the result of using a stepwise algorithm of adding and dropping terms by using the Akaike Information Criterion. The multiple R^2 for this least square fit is 0.8976, and the residual standard error is 0.1015. Table 5 gives the estimated coefficients, their standard errors and the corresponding *t*-statistics to test the partial effect of each variable. The plot of the residuals versus fitted values, Figure 10, suggests near homoscedasticity. In Figure 10, observations number 1, 66 and 98 appear to be outliers. The outliers do have two common characteristics. All three predictions have the variables 'mcrew' and 'rate' at the lowest level. This commonality among outliers suggests that the selected multiple regression model will predict poorly outside of the range of the input data. The outliers are a result of system overload. When 'rate' is at its lowest setting the time between maintenance actions is at a minimum. During this time of increased maintenance there are fewer maintainers to handle the workload and the system is more prone to developing a backlog of aircraft awaiting maintenance.

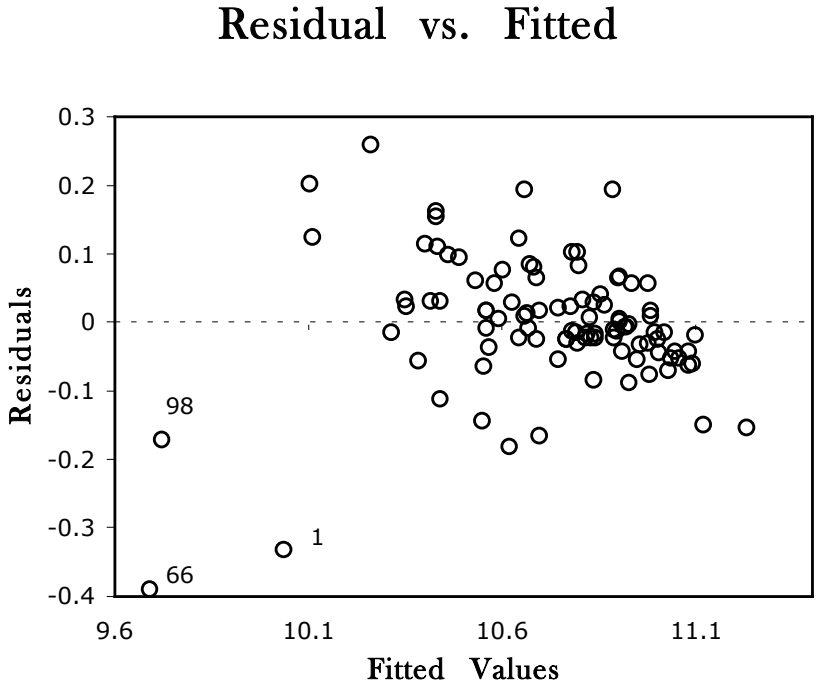

Figure 10. Residual plot of availability.

The main driver of availability is phase maintenance. Phases are a series of related inspections performed at specific intervals. For the H-53E the interval is 150 hours. The series consists of four inspections, labeled A through D. Each of the inspections is intended to be about the same length, though for the H-53E the C phase requires more than twice the man-hours of the others. To affect availability more significantly the phase interval would need to be lengthened or the time spent in phase shortened. While outside the scope of this study it is worth noting that IMD HUMS is expected to reduce the number of maintenance man-hours required in a phase inspection. Draper Labs (Davis, 1997) estimated that maintenance time would be reduced by around 20% (more or less depending on the particular phase). The estimates were based on the purported capabilities of IMD HUMS and a review of specific tasks required with each inspection phase.

C. RATIO OF FCF HOURS TO TOTAL HOURS

The multiple regression model of the ratio of FCF flight hours to total flight hours is estimated using a stepwise algorithm and the Akaike Information Criterion.

Table 6. Flight hour ratio regression details

Coefficients	Estimate	Std.Error	t-value	Pr(>|t|)
(Intercept)	0.326	0.005	71.633	<2e-16
Imds	-0.007	0.000	-14.298	<2e-16
mcrew	-0.001	0.001	-0.958	0.341
Fcrew	-0.003	0.001	-2.098	0.039
Rate	-0.144	0.003	-45.999	<2e-16
policy	0.004	0.003	1.545	0.126
Imds:rate	0.004	0.000	9.017	0.000
mcrew:fcrew	0.000	0.000	1.739	0.086
mcrew:policy	-0.001	0.000	-1.630	0.107
Imds:policy	0.000	0.000	1.398	0.165
F-statistic: 968.3 on 9 and 90 DF, p-value: < 2.2e-16				

This fit and its interpretation is similar to that of EMT. Increasing the number of IMD HUMS configured aircraft is the most significant factor, next to 'rate,' in decreasing the expected ratio. The multiple R^2 for this least square fit is 0.9898 and the residual standard error is 0.00234. Table 6 gives the estimated coefficients, their standard errors and the corresponding *t*-statistics to test the partial effect of each variable. Diagnostic plots of the residuals support the multiple regression modeling assumptions of Normal errors and homoscedasticity

To put the results in operational terms, if 'rate' and 'policy' have a value of 1.00, then for each additional aircraft with IMD HUMS the ratio of FCF time to total time is reduced by 0.23%. If a squadron of 12 aircraft and annual flight hour total of 2100 hours were equipped with IMD HUMS, that commander would expect an additional 58 flight hours for training and support that had been used for FCF.

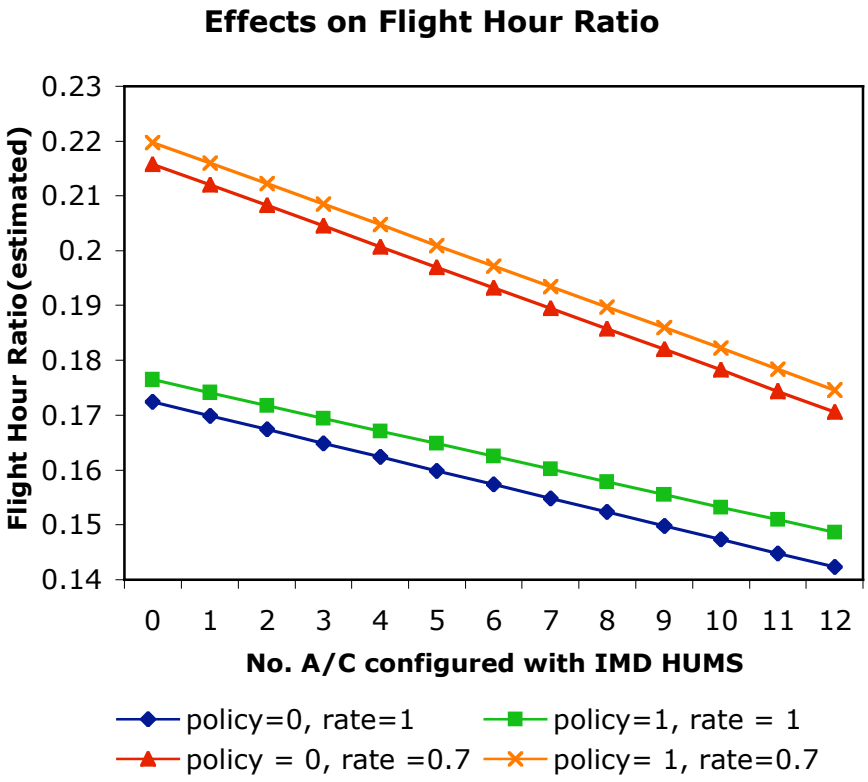

Figure 11. Effects of 'rate', 'policy', and 'imds' on Flight Hour Ratio (estimated)

As with EMT, the effect of increasing the number of aircraft configured with IMD HUMS depends on the 'rate' variable. When 'rate' is low, the time between maintenance actions is lower than normal; there are more maintenance actions and, therefore, more FCF's. In an environment with a greater number of FCF's the benefits of IMD HUMS are greater than normal. This interaction is illustrated in Figure 11. The effects of 'policy' are also portrayed in Figure 11. Under a policy that allows the maintainer to adjust the rotor system without the need for an FCF, there is some flight time recovered. The difference is small in an environment where maintenance is occurring more frequently. In a situation where maintenance is happening less often, the vibrations will have a chance to grow for a longer time and policy will have a greater effect. Additionally, if the true rate of vibration growth is greater than modeled in this run of the simulation then the effect of policy will be more pronounced. Using the output from the experiment conducted with EMT, Figure 12 compares the effect of vibration growth rate

under a policy that requires a full RTB in cases where recorded in-flight vibrations exceed a threshold.

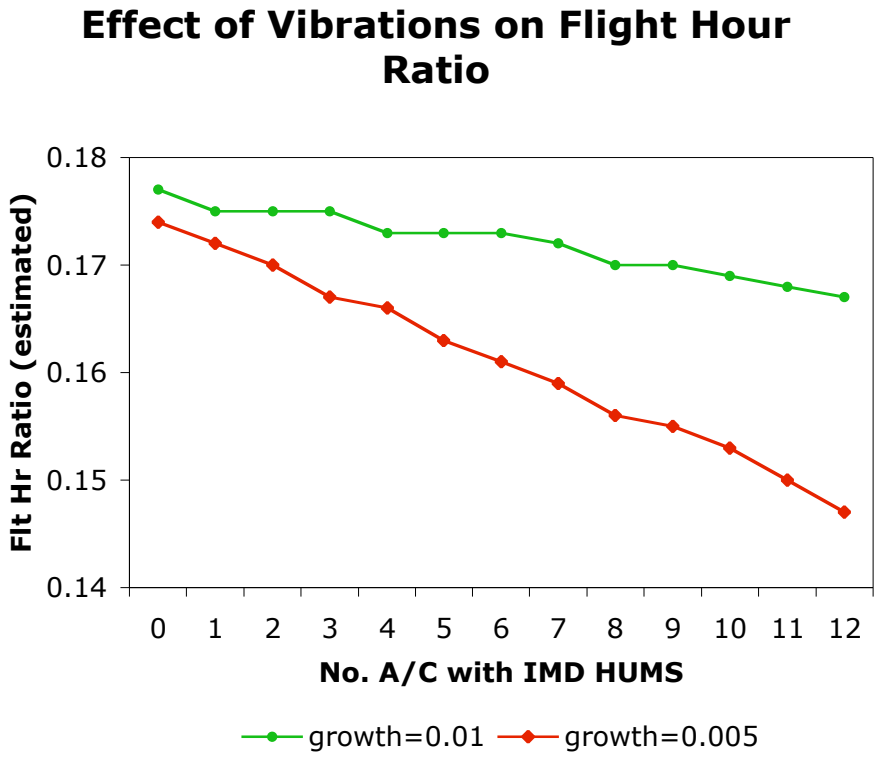

Figure 12. Effects of Vibration Growth on Expected Flight Hour Ratio (estimated)

As the legend indicates, the bottom line represents a vibration growth rate of 0.005 ips per flight hours. The top line is a plot of the ratio when the vibration growth rate is equal to 0.01 ips per flight hour. The standard errors for the expected ratio of flight hours of FCF hours to total hours are 0.011 or less for each point. If vibration growth rate is high then policy will make a difference.

THIS PAGE INTENTIONALLY LEFT BLANK

VI. CONCLUSIONS AND RECOMMENDATIONS

The results of the simulation suggest that maintenance man-hours can be recovered from RTB using IMD HUMS. Greater gains will likely be realized if maintenance policy is adjusted. The results also imply that while IMD HUMS does increase aircraft availability the net effect is quite small. Finally, the study indicates that by configuring fleet helicopters with IMD HUMS the Marine Corps can effectively increase the number flight hours used for flight operations other than FCF without additional annual flight hours. The outcomes of both EMT and flight hour ratio were affected by the maintenance policy under which the experiment was conducted. Both the Army and the Marine Corps have begun to adjust their practices to take advantage of the benefits of continuous monitoring while minimizing additional maintenance resulting from increased awareness.

To further take advantage of the system's capabilities considerations should be give to opportunistic RTB, in which a RTB smoothing event is performed when it is not required, but the opportunity to do so is presented as a result of the current operational tempo or other maintenance actions. The point of opportunistic RTB is to take acceptable vibrations to an even lower lever in order to capitalize on the advantages of a smoother aircraft.

When maintenance policy prevents maintainers from making small rotor adjustments outside of a FCF, vibration growth has the effect of lessening the benefits derived from the installation of IMD HUMS. While IMD HUMS is making RTB a more efficient process by requiring fewer adjustments, a high vibration growth rate combined with a strict maintenance policy that requires a full RTB should the recorded vibrations increase to a level greater than 0.3 ips will detract from the gains of using IMD HUMS.

Further enhancements to the simulation used for this study will provide a better estimation of aircraft availability. To accurately estimate A_O, the simulation will need to include a greater range of maintenance actions like, intermediate- and depot-level maintenance, along with a model for aviation supply and logistics. Finally, this model

does not increase the time between failures as a result of the decreased wear and damage due to flying a smoother aircraft. Further analysis could quantify these effects.

Follow-on studies that do not require a major modification to the simulation are also possible. The IMD HUMS system is expected to reduce the time required to complete phase maintenance and other fight hour based inspections. The 1997 Draper report (Davis, 1997) estimated the efficiencies expected to be gained during these inspections by the installation of IMD HUMS. The simulation used for this study can be used to estimate the effect on A_O. As data becomes available to validate the claims of the Draper report and the manufacturer's expectations, further analysis of the operational impact can be drawn. Another study of interest may be the effect of a high false alarm rate. False alarms are the result of setting thresholds for maintenance action too low. In addition to the rotor system, vibrations are monitored in the tail rotor drive shaft and the engines. The transition from inspection interval-based maintenance to continuous monitoring-based maintenance leaves room for future analysis, perhaps by this simulation model, of the effects of various maintenance policies.

APPENDIX A: MODEL INPUT AND OUTPUT

Table 7. WUC's

Count	WUC	FCF	RTB	Description
2036	22600	Yes	No	T64 ENGINE
231	22616	Yes	No	VARIABLE GEOMETRY ACTUATOR
117	2261500	Yes	No	VARIABLE GEOMETRY LINKAGE ASSEMBLY
371	22662	Yes	No	MAIN FUEL CONTROL
63	14E1510	Yes	No	UPPER INPUT ROD ASSEMBLY
82	14E1610	Yes	No	AFCS PITCH SERVO CYLINDER ASSEMBLY
67	14E1620	Yes	No	AFCS COLLECTIVE SERVOCYLINDER ASSEMBLY
58	14E1630	Yes	No	AFCS ROLL SERVOCYLINDER ASSEMBLY
61	14E1640	Yes	No	AFCS YAW SERVOCYLINDER ASSEMBLY
53	14E1810	Yes	No	LINEAR SERVO ASSEMBLY
67	14E1F3D	Yes	No	FOWARD/AFT SHAFT/BELLCRANK ASSEMBLY
70	14L1A10	Yes	No	TAIL ROTOR TANDEM SVOCYLINDER ASSEMBLY
67	14M1210	Yes	No	MN RTR SVOCYL CTRG SPR CYLINDER ASSY
525	14M1710	Yes	No	MAIN ROTOR TANDEM SERVOCYLINDER ASSY
3308	15A6100	Yes	Yes	MAIN ROTOR BLADE ASSEMBLY
54	15A6130	Yes	Yes	MAIN ROTOR BLADE SUBASSEMBLY
213	15A70	Yes	Yes	MAIN ROTOR HEAD ASSEMBLY
187	15A7400	Yes	Yes	MAIN ROTOR HEAD HUB ASSEMBLY
476	15A7420	Yes	Yes	MAIN ROTOR BLADE EXTENDER ASSEMBLY
114	15A7495	Yes	No	PITCH LOCK ASSEMBLY
1490	15A74A0	Yes	Yes	MAIN ROTOR HEAD CONTROL ROD ASSEMBLY
310	15A74A2	Yes	Yes	UPPER ROD END ASSEMBLY
2244	15A74J0	Yes	Yes	MN RTR HD HUB SLEEVE/SPINDLE ASSY
382	15A7500	Yes	Yes	MAIN ROTOR HEAD SWASHPLATE ASSY
86	15A7510	Yes	No	MN RTR HD ROTATING SCISSORS ASSY
197	15A7511	Yes	No	UPPER LINK ASSEMBLY
117	15A7512	Yes	No	LOWER LINK ASSEMBLY
797	15AD100	Yes	No	TAIL ROTOR BLADE ASSEMBLY
136	15AF400	Yes	Yes	HEAD/HUB ASSEMBLY
96	26D1100	Yes	Yes	MAIN GEAR BOX/ROTOR HEAD INSTL/ASSY
384	26D1170	Yes	Yes	MAIN GEAR BOX ASSEMBLY

Table 7 is a table of the WUC's used in the model. The count column is the number of inter-arrival times found from the fleet database from Jan 00 through Dec 03, inclusive. The count of inter-arrival times includes those with a zero value. Censored times were not included in the formulation of the distribution. Inter-arrival times were collected using an S-plus function written by Professor Sam Buttrey.

Table 8. Design Points generated from a Latin Hypercube

imds	mcrew	fcrew	rate	policy
8	5	2	0.7	1
11	9	2	0.9	0
10	8	2	0.7	0
7	6	2	1	0
9	9	3	1	1
3	9	2	1	0
10	9	3	0.9	1
3	9	2	0.9	0
12	8	1	1.1	0
6	10	3	1	1
1	8	3	1.1	1
10	7	2	1.1	1
11	9	3	0.8	0
10	8	1	0.7	1
11	6	2	1	1
7	10	3	1.1	0
10	6	2	1	0
5	7	3	0.8	0
1	5	4	0.8	1
5	6	1	0.8	0
12	7	3	1.1	1
6	9	3	0.9	1
6	8	1	1.1	1
7	6	1	0.8	0
8	5	3	0.9	1
0	6	2	1.1	0
6	9	3	0.9	1
2	7	4	1.1	0
9	7	3	0.9	1
8	8	3	1.1	1
3	6	4	1.1	0
12	9	3	1	1
8	7	4	0.9	0
1	9	2	1.1	1
11	9	2	1.1	0
6	8	3	0.8	1
5	5	4	1	1
2	10	1	1.2	1
4	8	2	1	0
4	8	3	1.1	1
12	7	2	0.9	0
8	10	2	1.1	1
5	7	1	1	1
2	8	1	0.9	0
4	7	4	0.9	1
2	8	2	0.9	1
6	8	2	1.2	0
4	6	1	0.8	0
8	9	4	1	0
6	8	3	0.8	1
3	10	3	0.8	0
0	8	2	0.9	0
8	10	2	0.9	1
8	8	1	0.9	0
5	7	2	1	0
5	7	3	1.2	0

2	6	4	0.8	1
1	9	4	1.2	0
9	7	2	1	1
4	7	1	0.7	0
4	9	1	0.8	1
10	9	2	1.2	0
9	10	4	0.7	1
1	8	1	1.2	0
9	9	3	0.9	0
0	5	1	0.7	1
4	6	2	0.8	1
9	9	3	1.1	0
3	6	3	1	0
3	5	3	0.8	0
3	7	1	0.7	1
11	9	1	0.8	1
11	8	2	1	0
4	6	2	1	1
0	10	3	1.1	0
5	10	3	1.2	0
7	6	4	1	0
7	5	2	1.2	1
7	7	2	0.8	1
0	6	3	0.7	1
5	6	4	0.8	1
2	7	4	0.9	0
6	5	2	0.9	0
9	8	3	0.8	1
11	7	2	1.1	0
4	9	3	1.1	0
12	6	3	1.2	1
2	8	3	0.8	0
3	5	4	0.8	0
11	6	2	1.1	1
1	6	4	0.9	1
9	6	4	1	0
8	8	2	0.9	1
7	6	2	1.1	1
7	9	3	1	1
10	7	3	1	0
10	10	4	1.2	0
1	5	1	0.7	1
2	7	3	0.8	1
1	7	2	0.7	0

imds Number of aircraft configured with IMD HUMS in a squadron of 12 aircraft

mcrew Number of maintenance crews, or simultaneous maintenance actions that can be in action.

fcrew Number of FCF crew assigned each day

rate Maintenance rate multiplier. Simulates increased or decreased maintenance as a result of the operating environment.

policy A policy value of '1' indicates a policy where a RTB is required if recorded in-flight vibrations are greater than 0.3 ips.

Table 9. Simulation Output

avail	availSD	fltHrs	fltHrsSD	fcfHrs	fcfHrsSD	tune	tuneSD	dmmh	dmmhSD	ratio	ratioSD
9.706	1.201	2867.213	35.461	557.323	35.318	2.14	2.204	1618.152	129.843	0.194	0.01
10.807	0.112	2739.026	30.396	429.026	30.396	0	0	1052.473	103.099	0.157	0.009
10.488	0.118	2842.842	37.474	532.842	37.474	0	0	1396.503	131.64	0.187	0.011
10.831	0.303	2737.369	29.91	427.369	29.91	0	0	1247.143	105.931	0.156	0.009
10.883	0.083	2732.577	34.129	422.577	34.129	20.04	12.501	1148.819	94.387	0.155	0.011
10.82	0.101	2780.672	39.952	470.672	39.952	0	0	1588.864	148.983	0.169	0.012
10.766	0.102	2755.689	28.24	445.689	28.24	10.92	8.673	1140.242	104.31	0.162	0.009
10.689	0.111	2814.689	31.988	504.689	31.988	0.02	0.141	1722.577	126.796	0.179	0.009
10.968	0.098	2680.346	22.608	370.346	22.608	0	0	866.253	75.817	0.138	0.007
10.838	0.099	2750.876	32.767	440.876	32.767	9.02	9.481	1343.264	136.588	0.16	0.01
10.924	0.091	2748.584	39.505	438.584	39.505	0.88	1.304	1569.01	158.057	0.159	0.012
10.992	0.098	2688.834	30.309	378.834	30.309	38.88	22.302	964.724	105.9	0.141	0.01
10.666	0.109	2778.566	30.509	468.566	30.509	0	0	1149.76	88.498	0.169	0.009
10.447	0.322	2844.968	29.085	534.968	29.085	2.34	2.685	1392.616	125.271	0.188	0.008
10.896	0.1	2718.67	27.535	408.67	27.535	29.3	21.443	979.372	80.334	0.15	0.009
10.971	0.095	2706.421	34.027	396.421	34.027	0.06	0.24	1178.435	121.683	0.146	0.011
10.878	0.094	2719.401	27.637	409.401	27.637	0	0	1057.746	92.271	0.15	0.009
10.591	0.104	2847.885	35.687	537.885	35.687	0	0	1727.349	148.992	0.189	0.01
10.382	0.35	2892.242	33.119	582.242	33.119	0.26	0.664	2099.381	139.931	0.201	0.009
10.584	0.125	2846.608	40.444	536.608	40.444	0	0	1708.579	155.342	0.188	0.012
10.982	0.081	2687.678	23.848	377.678	23.848	47.54	29.989	869.678	66.274	0.14	0.008
10.741	0.093	2785.057	27.832	475.057	27.832	6.12	5.812	1476.902	107.406	0.17	0.008
10.906	0.106	2727.737	31.013	417.737	31.013	18.2	19.135	1278.859	124.911	0.153	0.01
10.583	0.114	2826.321	33.502	516.321	33.502	0	0	1518.912	126.784	0.183	0.01
10.404	1.052	2770.89	32.591	461.88	31.107	9.44	7.995	1299.208	106.878	0.167	0.009
10.866	0.087	2758.013	36.173	448.013	36.173	0.06	0.24	1652.602	154.081	0.162	0.011
10.741	0.093	2785.057	27.832	475.057	27.832	6.12	5.812	1476.902	107.406	0.17	0.008
10.894	0.106	2746.69	41.836	436.69	41.836	0.06	0.24	1514.321	155.41	0.159	0.013
10.756	0.096	2768.542	26.863	458.542	26.863	9.08	8.843	1237.476	92.177	0.166	0.008
10.959	0.09	2712.142	29.641	402.142	29.641	20.02	13.543	1128.731	99.896	0.148	0.009
10.902	0.104	2731.269	31.491	421.269	31.491	0.12	0.328	1417.488	118.629	0.154	0.01
10.906	0.097	2712.418	29.168	402.418	29.168	30.74	20.869	919.088	77.997	0.148	0.009
10.764	0.108	2780.962	34.12	470.962	34.12	0	0	1346.942	131.122	0.169	0.01
10.924	0.091	2748.584	39.505	438.584	39.505	0.88	1.304	1569.01	158.057	0.159	0.012
11.002	0.084	2687.251	20.573	377.251	20.573	0	0	916.948	74.917	0.14	0.007
10.576	0.105	2837.062	37.198	527.062	37.198	2.88	4.034	1603.904	130.605	0.186	0.011
10.792	0.203	2758.568	32.731	448.568	32.731	5.42	5.183	1412.463	118.921	0.162	0.01
11.004	0.078	2710.815	31.817	400.815	31.817	3.12	3.088	1384.514	112.157	0.148	0.01
10.802	0.111	2774.953	32.973	464.953	32.973	0.04	0.198	1518.225	132.563	0.167	0.01
10.945	0.099	2736.574	36.285	426.574	36.285	8.52	7.731	1395.843	133.598	0.156	0.011
10.777	0.098	2735.956	24.816	425.956	24.816	0	0	988.002	88.103	0.156	0.008
10.958	0.102	2710.363	29.05	400.363	29.05	20.26	13.188	1111.077	106.397	0.148	0.009
10.85	0.108	2759.307	27.97	449.307	27.97	6.34	6.647	1397.796	130.741	0.163	0.008
10.676	0.129	2829.129	35.973	519.129	35.973	0	0	1813.122	162.103	0.183	0.01
10.714	0.113	2807.661	34.076	497.661	34.076	3.1	3.066	1635.145	150.203	0.177	0.01
10.679	0.11	2827.852	36.238	517.852	36.238	0.84	1.218	1807.203	130.95	0.183	0.01
11.04	0.08	2681.66	31.453	371.66	31.453	0.08	0.274	1129.515	105.059	0.138	0.01
10.514	0.201	2870.393	43.61	560.393	43.61	0	0	1859.914	178.309	0.195	0.012
10.884	0.097	2727.977	36.263	417.977	36.263	0.02	0.141	1174.41	125.928	0.153	0.011
10.576	0.105	2837.062	37.198	527.062	37.198	2.88	4.034	1603.904	130.605	0.186	0.011
10.53	0.102	2882.392	37.08	572.392	37.08	0	0	1933.957	141.82	0.198	0.01
10.621	0.098	2846.963	44.227	536.963	44.227	0.04	0.198	1983.657	184.391	0.188	0.013
10.753	0.094	2765.257	25.431	455.257	25.431	8.48	9.081	1280.778	104.794	0.165	0.008
10.764	0.105	2761.621	30.292	451.621	30.292	0	0	1283.502	124.491	0.163	0.009
10.84	0.129	2750.571	36.221	440.571	36.221	0	0	1390.294	127.848	0.16	0.011
11.032	0.097	2690.075	33.315	380.075	33.315	0.22	0.887	1190.415	122.769	0.141	0.011
10.473	0.238	2886.892	32.177	576.892	32.177	0.7	1.199	2022.996	118.629	0.2	0.009
10.976	0.095	2713.587	30.25	403.587	30.25	0.2	0.535	1428.215	134.759	0.149	0.009
10.88	0.119	2726.559	26.789	416.559	26.789	22.56	16.759	1129.072	99.391	0.153	0.008
10.378	0.106	2928.993	40.164	618.993	40.164	0	0	2055.26	162.948	0.211	0.011
10.55	0.111	2860.684	36.173	550.684	36.173	1.4	1.796	1821.465	127.361	0.192	0.01
11.081	0.082	2664.11	28.807	354.11	28.807	0.02	0.141	912.349	95.411	0.133	0.009

10.438	0.121	2861.628	27.808	551.628	27.808	1.82	2.768	1505.207	100.865	0.193	0.008
10.978	0.093	2714.729	32.581	404.729	32.581	0.12	0.328	1441.654	124.284	0.149	0.01
10.765	0.102	2755.434	29.017	445.434	29.017	0	0	1218.159	92.417	0.162	0.009
9.3	1.791	2966.034	44.62	657.574	44.528	0	0	2445.437	160.806	0.222	0.012
10.522	0.233	2856.238	41.739	546.238	41.739	1.18	1.612	1796.522	160.639	0.191	0.012
10.99	0.102	2693.54	27.783	383.54	27.783	0.02	0.141	1029.098	84.857	0.142	0.009
10.798	0.182	2776.044	37.439	466.044	37.439	0.02	0.141	1577.104	143.876	0.168	0.011
10.329	0.415	2863.011	36.013	553.011	36.013	0.02	0.141	1890.446	139.163	0.193	0.01
10.307	0.226	2936.422	39.856	626.422	39.856	0.44	0.611	2141.606	161.263	0.213	0.011
10.662	0.11	2780.203	34.831	470.203	34.831	6.5	5.814	1158.125	97.549	0.169	0.01
10.912	0.095	2716.374	25.553	406.374	25.553	0	0	1009.956	92.908	0.15	0.008
10.766	0.337	2773.469	32.081	463.469	32.081	4.2	5.047	1524.463	133.902	0.167	0.01
10.866	0.087	2758.013	36.173	448.013	36.173	0.06	0.24	1652.602	154.081	0.162	0.011
11.032	0.097	2690.075	33.315	380.075	33.315	0.22	0.887	1190.415	122.769	0.141	0.011
10.816	0.306	2735.735	26.707	425.735	26.707	0	0	1256.073	108.919	0.156	0.008
11.018	0.164	2686.315	24.538	376.315	24.538	29.76	20.677	1097.917	95.938	0.14	0.008
10.591	0.1	2824.527	29.796	514.527	29.796	2.68	2.853	1509.922	133.113	0.182	0.009
10.237	0.317	2975.23	41.248	665.23	41.248	0	0	2486.763	171.08	0.223	0.011
10.558	0.169	2851.214	33.414	541.214	33.414	2.66	2.528	1728.272	126.097	0.19	0.009
10.659	0.1	2827.659	43.196	517.659	43.196	0.04	0.198	1809.781	152.717	0.183	0.012
10.597	0.429	2787.937	30.789	477.937	30.789	0.02	0.141	1466.813	124.189	0.171	0.009
10.636	0.109	2808.182	31.558	498.182	31.558	4.82	6.137	1358.727	118.094	0.177	0.009
11.002	0.084	2687.251	20.573	377.251	20.573	0	0	916.948	74.917	0.14	0.007
10.91	0.09	2732.312	27.346	422.312	27.346	0.02	0.141	1384.422	126.584	0.154	0.008
11.076	0.081	2667.554	19.616	357.554	19.616	76.92	38.577	818.521	62.706	0.134	0.006
10.528	0.124	2881.969	37.225	571.969	37.225	0	0	2025.468	153.461	0.198	0.01
10.329	0.415	2863.011	36.013	553.011	36.013	0.02	0.141	1890.446	139.163	0.193	0.01
11.001	0.082	2696.762	24.061	386.762	24.061	48.4	29.566	939.813	87.81	0.143	0.008
10.655	0.106	2839.532	43.985	529.532	43.985	0.56	0.861	1901.324	171.004	0.186	0.013
10.893	0.086	2730.286	30.536	420.286	30.536	0.02	0.141	1145.432	101.385	0.154	0.009
10.753	0.094	2765.257	25.431	455.257	25.431	8.48	9.081	1280.778	104.794	0.165	0.008
10.963	0.114	2714.828	32.307	404.828	32.307	20.28	17.714	1186.742	109.093	0.149	0.01
10.867	0.079	2742.117	32.362	432.117	32.362	10.32	7.064	1265.889	115.63	0.157	0.01
10.888	0.084	2724.003	26.774	414.003	26.774	0	0	1057.063	95.681	0.152	0.008
11.077	0.087	2664.045	29.446	354.045	29.446	0.02	0.141	916.666	92.283	0.133	0.01
9.551	1.193	2962.435	46.511	653.645	44.193	0.08	0.274	2345.688	182.711	0.22	0.012
10.543	0.111	2880.292	38.037	570.292	38.037	0.54	0.788	1999.292	159.488	0.198	0.011
10.297	0.181	2953.28	37.935	643.28	37.935	0	0	2313.598	128.475	0.218	0.01

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX B. SIMULATION CODE

The java classes Helo, Operations, Maintenance and Functional Check Flight, the core elements of the simulation, are included in this section. A copy of the latest version of Simkit is available on the World Wide Web at the URL http://diana.or.nps.navy.mil/Simkit

1. Helo class

```
package rtb;

import simkit.*;
import simkit.random.*;
import java.util.*;
import java.io.*;
import java.net.*;
import java.text.DecimalFormat;

/*
 * File: Helo.java
 * Created: January 5, 2004, 11:07 AM
 */

/**
 * <p>This class will carry the aircraft attributes and Random Variates.
 .</p>
 * @author Mark Revor
 */
public class Helo extends SimEntityBase{
  // class constants
  private static DecimalFormat df= new DecimalFormat("##.###");

  // class variables
  // instance variables
  private int sideNumber;
  private double aFH; //Airframe Hours
  private HashMap nextFail; //A HashMap of Doubles indicating the time until the next failure
  private HashMap nextFailYellow; //A Hashmap of Doubles 95% of the time to next fail
  private HashMap ttf; //HashMap of RandomVariates
  private HashMap ttfFCF; //HashMap of FCF related details
  private HashMap ttr; //HashMap of RandomVariates
  private HashMap rtbP;//HashMap of Doubles.  Proportion wuc is a group that require an RTB
  private HashMap fcf; //HashMap of RandomVariates
  private HashMap maint;
  private HashMap condMaint;
  private int phase; //next phase type (A=1, B=2, C=3, D=4)
  private double gndAdj; //number of adjustments made to the ground track
  private double fltAdj; // number of adjustments made to the flight track
  private LinkedList wuc; // Work Unit Code, a string to describe the maintenace
  private boolean up; //either up or down
  private boolean needFCF;
  private boolean needRTB;
  private boolean gndTrack; // true- ground track is in
  private double receivedMAF; //time a MAF was received
  private double inworkMAF; //time maintenance began on a MAF
  private double lastRTB; //airframe flight hours at last RTB
  private double vibeLevel;  //level of vibration at last RTB
  private double vibeLimit;  // max vibes before RTB is required
  private double vibeGrowth; //rate at which vibes increase with flight hours
  private boolean needTuning; //aircraft needs tuning


  // class methods
```

```
        // constructor methods

        /**
         * Constructor.
         */
        public Helo(HashMap f, HashMap fFCF, HashMap r, HashMap rtbPRV, HashMap fcfRV,
RandomVariate tillPhase, int phase, int side) {
          sideNumber = side;
          aFH = 0.0;
          ttf = (HashMap) f.clone();
          ttfFCF = (HashMap) fFCF.clone();
          nextFail = new HashMap();
          nextFailYellow = new HashMap();
          fcf = (HashMap)fcfRV.clone();
          for (Iterator i = ttf.keySet().iterator(); i.hasNext(); ) {
            Object key = i.next();
            Object rv = ttf.get(key);
            setNextFail(key, rv);
          }
          for (Iterator i = ttfFCF.keySet().iterator(); i.hasNext(); ) {
            Object key = i.next();
            setNextFail((String)key, (RandomVariate)ttfFCF.get(key));
          }
          ttr = (HashMap) r.clone();
          rtbP = (HashMap)rtbPRV.clone();
          this.phase = phase;
          setUp(true);
          setNextFail("phase", tillPhase);
          wuc = new LinkedList();
          setNeedFCF(false);
          setNeedRTB(false);
          vibesAfterRTB();
          //System.out.println(vibeLevel);
          if(fcf.containsKey("vibeGrowth")){
             vibeGrowth = ((Double)fcf.get("vibeGrowth")).doubleValue();
          }
          else{
             vibeGrowth = 0.005;
          }
          if(fcf.containsKey("vibeLimit")){
             vibeLimit = ((Double)fcf.get("vibeLimit")).doubleValue();
          }
          else{
             vibeLimit = 0.6;
          }
          setNeedTuning(false);
          maint = new HashMap();
          condMaint = new HashMap();
        }

        // instance methods
        public void reset(){
          setUp(true);
          setNeedFCF(false);
          setNeedRTB(false);
          //this.log('r', 0.0);
```

```
}
//Status methods

/**
 *Checks for pre-flight and in-flight failures based on the number of pending
 *post-flight maintenance actions.  The method has a conditional statement for
 *including maintenance that is about to happen.  An arrival multiplier is
 *set in the "setNextFail" method.
 */
public int preFlight(double length){
  boolean boo = false;
  for (Iterator i = nextFail.keySet().iterator();i.hasNext();){
    Object key = i.next();
    if (aFH+length >= ((Double)nextFail.get(key)).doubleValue()){
       boo = true;
    }
  }
  if (boo){
    for (Iterator i = nextFailYellow.keySet().iterator();i.hasNext();){
      Object key = i.next();
      if(aFH+length >= ((Double)nextFailYellow.get(key)).doubleValue()){
          wuc.add(key);
       }
    }
  }
  if(((RandomVariate)fcf.get("dice")).generate() <  (wuc.size()/200)){
    return 1; //probability of a pre-flight abort based on the size of the Wuc list
  }
  else if(((RandomVariate)fcf.get("dice")).generate() < (wuc.size()/100)){
    return 2; //Probability of an in-flight abort.
  }
  else
    return 3;
}
/**
 * Returns the status (Up or Down) of the aircraft after a flight
 */
public boolean postFlightStatus(){
  if(wuc.size()>0){
    return false;
  }
  else{
    return true;
  }
}

/**
 * Method for determining the repair time.  Repair time based on the
 * type (WUC) associated with the failure.  Accounts for multiple maintenace
 * actions from the same group.
 */
public double endRepairTime(){
  Object key = wuc.getFirst();
  double time = 0.0;
  double repeat=1.0;
  if((String)key == "phase"){
```

```
      key =((String)key+phase);
    }
    if(ttr.containsKey("Mult"+key)){
      repeat = ((RandomVariate)ttr.get("Mult"+key)).generate();
    }
    long loop=Math.round(repeat);
    for(int i=0;i<loop;i++){
      time = time+((RandomVariate)ttr.get(key)).generate();
    }
    return time;
  }

  public int endRepairStatus(){
    int end = 0;
    Object key = wuc.removeFirst();
    if(ttfFCF.containsKey(key)){
      this.setNeedFCF(true);
      this.setNextFail(key, ttfFCF.get(key));
      if(!needRTB){
        this.setNeedRTB(this.doesItNeedaRTB(key));
      }
    }
    else {
      this.setNextFail(key, ttf.get(key));
    }
    if(!needFCF && wuc.size()==0){
      this.setUp(true);
      end = 1;
    }
    else if(wuc.size()>0){
      this.setUp(false);
      end = 2;
    }
    else if(needFCF && wuc.size()==0){
      this.setUp(false);
      end = 3;
    }
    return end;
  }

  /**
   *Developed for grouped WUC's.  If a group contains entirely codes that require a
   *RTB then the rtbP value will be one.  If the opposite is true the value will be zero
   *and if there is a mix then the value is the proportion that requires an RTB
   */
  public boolean doesItNeedaRTB(Object key){
     if (((RandomVariate)fcf.get("dice")).generate() < ((Double)rtbP.get(key)).doubleValue()){
        setNeedRTB(true);
        return true;
     }
     else{
        return false;
     }
  }

  /**
```

```
 *Method to facilitate the tracking of the number of FCF MAFs and RTB MAFs
 *completed in the maintenance class
 */
public double[] jobCount(){
  double[] count = {0.0, 0.0};
  Object key = wuc.getFirst();
  if(ttfFCF.containsKey(key)){
     count[0] = 1.0;
     count[1] =((Double)rtbP.get(key)).doubleValue();
  }
  return count;
}

/**
 * Sets the number of adjustments required to track and balance the Helo.
 * Set and used in the Functional Check Flight class
 */
public void tracknBalanceRuns(){
  fltAdj= Math.round(((RandomVariate)fcf.get("fltAdj")).generate());
  if(this.getNeedTuning()){
     fltAdj=Math.ceil(fltAdj/2.0);
  }
}

public void gndTrackRuns(){
  gndAdj=Math.round(((RandomVariate)fcf.get("gndAdj")).generate());
  if(this.getNeedTuning()){
     gndAdj = Math.ceil(gndAdj/2.0);
  }
}

public void setUp(boolean status) {up = status;}

public boolean getUp(){return up;}

public boolean getNeedFCF() {return needFCF;}

public boolean getNeedRTB() {return needRTB;}

public void setNeedFCF(boolean b) {needFCF = b;}

public void setNeedRTB(boolean b) {needRTB = b;}

/*Airframe flight hours methods*/
public double getAFH(){return aFH;}

public void setAFH(double flightHours) {aFH = flightHours;}

public void updateAFH(double flightTime){
  aFH = aFH + flightTime;
}

/*maintenance related methods*/

public void setNextFail(Object key, Object rv) {
  double tempo = ((Double)fcf.get("tempo")).doubleValue();
```

```
    double time = tempo*((RandomVariate)rv).generate();
    nextFail.put(key, new Double(aFH+time));
    nextFailYellow.put(key, new Double(1*time+aFH)); //change 1 to .95 (etc) to include nearly
failed
  }

  public HashMap getNextFail(){return (HashMap)nextFail.clone();}

  public int getPhase() {return phase;}

  public void setPhase(int phase){
    this.phase = phase;
  }

  public double getReceivedMAF() {return receivedMAF;}

  public void setReceivedMAF(double time){
    receivedMAF = time;
  }

  public double getInworkMAF() {return inworkMAF;}

  public void setInworkMAF(double time){
    inworkMAF = time;
  }

  public List getWUC() {return (LinkedList)wuc.clone();}

  public String getCurrentWUC(){
     return (String)wuc.getFirst();
  }

  public void clearWUCs() {wuc.clear();}

  public void condWucs(double[][] table){
    for(int i=0;i<wuc.size();i++){
       if(ttfFCF.containsKey(wuc.get(i)) && (String)wuc.get(i) != "phase"){
         maint.put(wuc.get(i), wuc.remove(i));
       }
    }
    for(Iterator i = maint.keySet().iterator(); i.hasNext();){
      Object key = i.next();
      int row = Integer.parseInt((String)maint.get(key));
      for(int j=0; j<table[row].length;j++){
        if(((RandomVariate)fcf.get("dice")).generate() < table[row][j]){
          condMaint.put(String.valueOf(j), String.valueOf(j));
        }
      }
    }
    maint.putAll(condMaint);
    for(Iterator i = maint.keySet().iterator(); i.hasNext();){
      wuc.addLast(maint.get(i.next()));
    }
    maint.clear();
    condMaint.clear();
  }
```

```
public double getGndAdj() {return gndAdj;}

public void setGndAdj(double i) {gndAdj = i;}

public double getFltAdj() {return fltAdj;}

public void setFltAdj(double i) {fltAdj = i;}

public boolean getGndTrack() {return gndTrack;}

public void setGndTrack(boolean boo) {gndTrack = boo;}

public double gndPrep(){
  return ((RandomVariate)fcf.get("gndPrep")).generate();
}

public double gndCollectTime(){
  return ((RandomVariate)fcf.get("tGndC")).generate();
}

public double adjustTime() {
  return ((RandomVariate)fcf.get("tAdj")).generate();
}

public double fcfNoRTB(){
  return ((RandomVariate)fcf.get("tFCFnoRTB")).generate();
}

public double fcfTime(){
  return ((RandomVariate)fcf.get("tFCF")).generate();
}

public double derigTime(){
  return ((RandomVariate)fcf.get("tDerig")).generate();
}

public double randomNum(){
  return ((RandomVariate)fcf.get("dice")).generate();
}

public void setLastRTB(double time){
  lastRTB = time;
}

public double getLastRTB(){return lastRTB;}

public void vibesAfterRTB(){
   vibeLevel=((RandomVariate)fcf.get("vibes")).generate();
}

/**
 *Determines the need for a tuning event.  Returns false if tuning is not needed.
 *Returns a true if tuning is needed.  Sets needRTB to true
 */
public void vibeUpdate(){
```

```
            double currentVibes= vibeLevel+(aFH-lastRTB)*vibeGrowth +
((RandomVariate)fcf.get("vibeNoise")).generate();
            if(currentVibes > vibeLimit){
               this.setNeedRTB(true);
               this.setNeedTuning(true);
            }
            else{
               this.setNeedTuning(false);
            }
         }

         public boolean getNeedTuning() {return needTuning;}

         public void setNeedTuning(boolean need){needTuning = need;}

         public int imds(){
           if(fcf.containsKey("IMDS")){
             return 1;
           }
           else {
             return 2;
           }
         }

         /**
          * An optional Logbook function.  If used, it will create a flight and
          * maintenance logbook for each aircraft
          */
         public void log(char x, double time){
            String file = "log"+sideNumber+".out";
            try {
              File parentDir = new File(rtb.Helo.class.getResource(".").getFile());
              File outFile = new File(parentDir, file);
              BufferedWriter writer = new BufferedWriter(new FileWriter(outFile, true));
              if(x=='w'){
                for(int i=0; i<wuc.size(); i++){
                  writer.write(sideNumber+", "+df.format(aFH)+", "+wuc.get(i)+"\n");
                  writer.flush();
                }
              }
              else if(x=='r'){
                 int i = 0;
                 writer.write("-------Reset-------------------------"+i+"\n");
                 writer.flush();
                 i++;
              }
              else{
                 writer.write(sideNumber+", "+df.format(aFH)+", 2K2, "+time+"\n");
                 writer.flush();
              }
                 } catch (Exception e) {
                    System.out.println("Problem serializing: " + e);
                 }
         }

         public String toString(){
```

```
    return sideNumber + ": Airframe flight hours: " + df.format(aFH);
  }

}
```

2. Operations Class

```
package rtb;
import simkit.*;
import simkit.Schedule;
import java.util.*;
import simkit.random.*;

/*
 * File: Operations.java
 * Created: January 5, 2004, 12:00 PM
 */

/**
 * <p>Recevies a fligth schedule and executes it.  Holds the queue for Up aircraft.</p>
 * @author Mark Revor
 */
public class Operations extends SimEntityBase {
  // class constants
  // class variables
  // instance variables
  protected int numberFlights;
  protected int numberUpAircraft;
  private int totalNumberAircraft;
  protected double totalFlightTime;
  protected LinkedList qUp;
  protected int dropped;
  protected int tunes;
  protected boolean policy; //true for RTB when tuning with HUMS
  protected double[][] cond;

  // class methods
  // constructor methods

  /**
   * Constructor.
   *@param int
   *    Number of aircraft in the squadron
   */
  public Operations(int aircraft, boolean policy, double[][] cond) {
    totalNumberAircraft = aircraft;
    qUp= new LinkedList();
    this.cond=cond;
    this.policy= policy;
  }

  // instance methods
  public void reset(){
    super.reset();
    numberUpAircraft= 0;
    totalFlightTime= 0.0;
    numberFlights = 0;
    dropped = 0;
    tunes=0;
    qUp.clear();
  }
```

```
/**
 * Initializes property change states.
 */
public void doRun() {
  firePropertyChange("numberUpAircraft", numberUpAircraft);
  firePropertyChange("flightTime", totalFlightTime);
  firePropertyChange("numberOfFlights", numberFlights);
  firePropertyChange("Tuning", tunes);
}

/**
 * Receives a schedule from and extracts information required for executing
 * the schedule.  Initiates a TakeOff event
 * @param FlightSchedule
 *    input is a FlightSchedule object
 */
public void doSchedule(FlightSchedule sched){
  HashMap event = new HashMap();
  double takeOff = -1.0;
  int events = sched.getNumberOfEvents();
  LinkedList eventList = sched.getSched();
  for(int i =0; i<events; i++) {
    event = (HashMap)eventList.removeFirst();
    waitDelay("TakeOff", ((Double)event.get("takeOff")).doubleValue(), event);
  }
}

/**
 * When TakeOff is called this method checks to see if there are enought aircraft
 * to launch the event.  It will launch as many as it can vice dropping the
 * entire event if the whole thing cannot be executed.  The method will tally
 * dropped flights and schedule a land event at the appropriate time.  Aircraft
 * hours are also incremented here.
 *@param event
 *    Take a Hashmap with "lengthOfFLight" and "aircraft" keys.
 */
public void doTakeOff(HashMap event) {
  double length = ((Double)event.get("lengthOfFlight")).doubleValue();
  int aircraft = ((Integer)event.get("aircraft")).intValue();
  if(aircraft > qUp.size()){
    dropped = dropped +(aircraft - qUp.size());
    //System.out.println((aircraft- qUp.size())+"dropped events at "+ Schedule.getSimTime());
    aircraft = qUp.size();
  }
  for (int i=0; i<aircraft; i++){
    Helo h = (Helo)qUp.removeFirst();
    switch (h.preFlight(length)){
      case 1:
        length = 0.0;
        waitDelay("Land", 0.0,h);
        break;
      case 2:
        length = length * h.randomNum();
        firePropertyChange("numberOfFlights", numberFlights, ++numberFlights);
        waitDelay("Land", length, h);
```

```
            break;
          case 3:
            firePropertyChange("numberOfFlights", numberFlights, ++numberFlights);
            waitDelay("Land", length, h);
            break;
          default: throw new IllegalArgumentException("invalid preflight code");
        }
        firePropertyChange("flightTime", length);
        totalFlightTime = totalFlightTime+length;
        h.updateAFH(length);
      }
    }

    /**
     * Uses postFlightStatus, a Helo method, to determine whether the aircraft should
     * be returned to the Up queue.  vibeUpdate is also called to determine the rotor vibe
     * levels are high enough to require rotor tuning.  If the aircraft does not need
     * tuningis not up then the "Down" event
     * is called.
     *@param a Helo object
     */
    public void doLand(Helo a){
      a.vibeUpdate();
      if(!a.postFlightStatus()){  //true means that there have been no fails during the flight
        a.setNeedTuning(false);
        //a.condWucs(cond);  //available using conditional relationshipe between WUCs
        //char w;//Uncomment this and next line to use Logbook function
        //a.log('w', 0.0);
        waitDelay("Down", 0.0, a);
      }
      else if(a.getNeedTuning()){
         if(a.imds()==2 || policy){ //imd = 2 means vats and policy = false says IMD a/c don't need
tune fcfs
           firePropertyChange("Tuning", tunes, ++tunes);
           firePropertyChange("numberUpAircraft", numberUpAircraft, --numberUpAircraft);
           waitDelay("FCF", 0.0, a);
         }
         else{
           qUp.add(a);
         }
      }
      else{
        qUp.add(a);
      }
    }

    /**
     *This method add a helo to the Up queue and increments the number of available
     *aircraft.  This method is only used for a down aircraft returning to an up
     *status or on initialization
     *@param a Helicopter object
     */
    public void doUP(Helo h){
      qUp.add(h);
      firePropertyChange("numberUpAircraft", numberUpAircraft, ++numberUpAircraft);
      h.setNeedFCF(false);
```

```
    h.setNeedRTB(false);
    h.setUp(true);
  }

  /**
   *This method add a helo to the decrements the number of available aircraft
   *@param a Helicopter object
   */
  public void doDown(Helo h){
    if(h.getUp()){
      firePropertyChange("numberUpAircraft", numberUpAircraft, --numberUpAircraft);
      h.setUp(false);
    }
  }

  /**
   * Returns the number of aircraft that have flown.
   *@return numberFlights
   */
  public int getNumberFlights() {return numberFlights;}

  /**
   * Returns the Total squadron flight time
   *@return total flight time
   */
  public double getTotalFlightTime() {return totalFlightTime;}

  /**
   * Returns the number of dropped flights.
   *@return dropped
   */
  public int getDropped() {return dropped;}

  /**
   * To string method
   *@return String output of total flight time, number of dropped flights and
   *the total number of aircraft.
   */
  public String toString(){
    return "Operations Summary\n\tOperational Flight hour total: " +totalFlightTime
      +"\n\tNumber of Dropped Flights: "+ dropped
      +"\n\tNumber of aircraft: " +totalNumberAircraft
      +"\n\tNumber of tuning events: "+tunes;
  }
}
```

3. Maintenance Class

```
package rtb;
import simkit.*;
import java.util.*;

/*
 * File: Maintenance.java
 * Created: January 5, 2004, 12:08 PM
 */

/**
 * <p>Methods for maintenance portion of the squadron.</p>
 * @author mrevor
 */
public class Maintenance extends SimEntityBase{
  // class constants
  // class variables
  // instance variables
  protected int numberAvailMaintCrews;
  protected int numberJobsComplete;
  protected double numberFCFJobs;
  protected double numberRTBJobs;
  protected double maintTime;
  private double timeComp;
  private int totalNumberMaintCrews;
  private LinkedList qMaint;

  // class methods
  // constructor methods

  /**
   * Constructor
   *@param
   *    Number of maintenance crews.  Note: not the number of maintainters.
   */
  public Maintenance(int crews, double timeComp) {
    totalNumberMaintCrews = crews;
    qMaint = new LinkedList();
    this.timeComp = timeComp;
  }

  // instance methods
  public void reset(){
    super.reset();
    numberAvailMaintCrews = totalNumberMaintCrews;
    numberJobsComplete = 0;
    numberFCFJobs = 0;
    numberRTBJobs = 0;
    qMaint.clear();
  }

  public void doRun() {
    firePropertyChange("maintCrews", numberAvailMaintCrews);
    firePropertyChange("numberInMaintQ", qMaint.size());
  }
```

```
    /**
     *Adds the downed helo to the FIFO maintenance queue.  If there are crews available
     *maintenance will begin
     */
    public void doDown(Helo h){
      qMaint.add(h);
      h.setReceivedMAF(Schedule.getSimTime());
      firePropertyChange("numberInMaintQ", qMaint.size()-1, qMaint.size());
      if (numberAvailMaintCrews >0){
        waitDelay("StartMaint", 0.0);
      }
    }

    /**
     *Gets the maintenance time required from Helo class using endRepairTime method
     *Also fires property changes for number of available crews and size of the queue
     */
    public void doStartMaint() {
      Helo h = (Helo)qMaint.removeFirst();
      h.setInworkMAF(Schedule.getSimTime());
      firePropertyChange("maintCrews", numberAvailMaintCrews, --numberAvailMaintCrews);
      firePropertyChange("numberInMaintQ", qMaint.size()+1, qMaint.size());
      waitDelay("EndMaint", h.endRepairTime()/timeComp, h); //this will depend on the type of
maintenance to be performed
    }

    /**
     *After maintenance is complete this class fires property changes for number of available
     *maintenance crews and records the completion time.  The next fail time is set and the
     *aircraft leaves here up or requiring an FCF
     */
    public void doEndMaint(Helo h) {
      double[] count = {0.0,0.0};
      firePropertyChange("maintCrews", numberAvailMaintCrews, ++numberAvailMaintCrews);
      maintTime = maintTime + (Schedule.getSimTime()- h.getInworkMAF());
      numberJobsComplete++;
      count = h.jobCount();
      //System.out.println(count[0]);
      //System.out.println(count[1]);
      numberFCFJobs += count[0];
      numberRTBJobs += count[1];
      //System.out.println(h.getCurrentWUC());
      switch(h.endRepairStatus()){
         case 1: waitDelay("UP", 0.0, h);
           //System.out.println("case 1");
           break;//no other maint, no FCF required
         case 2: waitDelay("Down", 0.0, h);
           //System.out.println("Case 2");
           break; //more maint required
         case 3: waitDelay("FCF", 0.0, h);
           //System.out.println("Case 3, rtb= "+h.getNeedRTB());
           break; //no other maint, FCF required
         default: throw new IllegalArgumentException("invalid endMaint code");
      }
    }
```

```
  /**
   *String output of the number of maintenance crews, jobs complete and time spent
   */
  public String toString(){
     return "Maintenance Summary:\n\tNumber of Maintenance Crews: "+totalNumberMaintCrews
        +"\n\tTotal number of jobs completed: "+numberJobsComplete
        +"\n\tTotal number of FCF jobs: "+numberFCFJobs
        +"\n\tTotal number of RTB jobs: "+numberRTBJobs
        +"\n\tTime in Maintenance: "+maintTime;
  }

}
```

4. Functional Check Flight Class

```
package rtb;

import java.util.*;
import simkit.*;
import simkit.random.*;
import java.io.*;
/*
 * File: RotorTracknBalance.java
 * Created: January 5, 2004, 11:49 AM
 */

/**
 * <p>Rotor Track and Balance details, maint and verification.</p>
 * @author Mark Revor
 */
public class FunctionalCheckFlight extends SimEntityBase{
    // class constants
    // class variables
    // instance variables
   private int fcfCrews;
   private int vats; // number of VATS/ATABS available to a squadron
   private int optTrack; //number of optical trackers available to a squadron
   protected int numberAvailVATS;
   protected int numberAvailOptTrack;
   protected int numberAvailFCFCrews;
   protected int numberRTB;
   protected LinkedList qFCF;
   protected LinkedList delayfcfQ;
   protected double rotorManHours;
   protected double fcfTime;
   protected int numberACFCFs; //number of aircraft needing an FCF
   protected int flights; // number of FCF flights
   protected int numberTunes; //number of Tuning events
    // class methods
    // constructor methods

    /**
     * Constructor.  Calls on the Helo class for service times.  The number of
     *crews available for test is set here.
     */
    public FunctionalCheckFlight(int crews, int vats, int optTrack) {
      fcfCrews = crews;
      this.vats = vats;
      this.optTrack = optTrack;
      qFCF = new LinkedList();
      delayfcfQ = new LinkedList();
    }

    // instance methods
    public void reset() {
      super.reset();
      numberAvailFCFCrews = fcfCrews;
      numberAvailVATS = vats;
      numberAvailOptTrack = optTrack;
```

```
    numberRTB = 0;
    rotorManHours = 0;
    fcfTime = 0;
    numberACFCFs = 0;
    flights = 0;
    qFCF.clear();
    delayfcfQ.clear();
  }

  public void doRun() {
    firePropertyChange("fcfCrew", numberAvailFCFCrews);
    firePropertyChange("vats", numberAvailVATS);
    firePropertyChange("OptTrack", numberAvailOptTrack);
    firePropertyChange("numberInFCFq", qFCF.size());
    firePropertyChange("numberFCFflights", flights);
    firePropertyChange("fcfTime", fcfTime);
    firePropertyChange("rotorMaint", rotorManHours);
  }

  /**
   *Removes a Helo from the FCF queue and fires a property change.  If crews are
   *available it calls for the start of an FCF with no delay.
   */
  public void doFCF(Helo h) {
    //System.out.println("FCF");
    qFCF.add(h);
    numberACFCFs++;
    firePropertyChange("numberInFCFq", qFCF.size()-1, qFCF.size());
    delayfcfQ.add(new Double(Schedule.getSimTime()));
    if(numberAvailFCFCrews >0) {
      waitDelay("StartFCF", 0.0);
    }
  }

  /**
   *This method determines whether or not the aircraft requires a RTB.  If it
   *does require an RTB the number of adjustments required are set by using the
   *gndTrackRuns  and tracknBalanceRuns methods of the Helo class.  Prep time is
   *recorded and start flight or start ground is called depending on the need
   *for an FCF.
   */
  public void doStartFCF(){
    //System.out.println("startFCF");
    for(Iterator i = qFCF.iterator(); i.hasNext();){
      Helo h = (Helo)i.next();
      if(!h.getNeedRTB()){
        i.remove();
        h.setGndAdj(0);
        h.setFltAdj(0);
        firePropertyChange("fcfCrew", numberAvailFCFCrews, --numberAvailFCFCrews);
        firePropertyChange("numberInFCFq", qFCF.size()+1, qFCF.size());
        firePropertyChange("DelayInFCFQ",                                    Schedule.getSimTime()-
((Double)delayfcfQ.removeFirst()).doubleValue());
        waitDelay("StartTestFlight",0.0, h);
      }
      else if (h.imds()==1 && h.getNeedTuning()){
```

```
                i.remove();
                firePropertyChange("numberInFCFq", qFCF.size()+1, qFCF.size());
                firePropertyChange("fcfCrew", numberAvailFCFCrews, --numberAvailFCFCrews);
                firePropertyChange("DelayInFCFQ",                    Schedule.getSimTime()-
((Double)delayfcfQ.removeFirst()).doubleValue());
                waitDelay("RTB", 0.0, h);
              }
              else if(h.imds()==1 && numberAvailOptTrack >0){
                i.remove();
                firePropertyChange("OptTrack", numberAvailOptTrack, --numberAvailOptTrack);
                firePropertyChange("numberInFCFq", qFCF.size()+1, qFCF.size());
                firePropertyChange("fcfCrew", numberAvailFCFCrews, --numberAvailFCFCrews);
                firePropertyChange("DelayInFCFQ",                    Schedule.getSimTime()-
((Double)delayfcfQ.removeFirst()).doubleValue());
                waitDelay("RTB", 0.0, h);
              }
              else if(h.imds()==2 && numberAvailVATS >0){
                i.remove();
                firePropertyChange("vats", numberAvailVATS, --numberAvailVATS);
                firePropertyChange("numberInFCFq", qFCF.size()+1, qFCF.size());
                firePropertyChange("DelayInFCFQ",                    Schedule.getSimTime()-
((Double)delayfcfQ.removeFirst()).doubleValue());
                waitDelay("RTB", 0.0, h);
              }
              else{
                  //do nothing and skip it
              }
            }
            //System.out.println("\t\t\t\t\tend of startFCF");
          }

          public void doRTB(Helo h){
            h.gndTrackRuns(); //set the number of adj needed to pass grnd check
            h.tracknBalanceRuns(); //sets the number of adj needed to pass RTB (in flight)
            double maintTime = h.gndPrep();
            //firePropertyChange("fcfCrew", numberAvailFCFCrews, --numberAvailFCFCrews);
            //firePropertyChange("numberInFCFq", qFCF.size()+1, qFCF.size());
            waitDelay("StartGroundTest", maintTime, h);
            numberRTB++;
            firePropertyChange("rotorMaint", maintTime);
            rotorManHours= rotorManHours+maintTime;
          }

          /**
           *Calls for the end of the ground test at a random time set by
           *tGndC random variae in the Execution class.
           */
          public void doStartGroundTest(Helo h){
            waitDelay("EndGround", h.gndCollectTime(), h);
          }

          /**
           *Checks to see if the required number of ground adjustments have been
           *completed.  If so, a flight is scheduled with no delays.  If not and adjustment
           *is made and start ground is called again.
           */
```

```
public void doEndGround(Helo h){
  double adjustTime = 0.0;
  if(h.getGndAdj() <= 0.0){
    waitDelay("StartTestFlight", 0.0, h);
  }
  else{
    h.setGndAdj((h.getGndAdj()-1.0));
    adjustTime = h.adjustTime();
    waitDelay("StartGroundTest", adjustTime,h);
    firePropertyChange("rotorMaint", adjustTime);
    rotorManHours = rotorManHours + adjustTime;
  }
}

/**
 *Checks to see what type of FCF it is (RTB or no RTB) and then schedules
 *the flight time accordingly
 */
public void doStartTestFlight(Helo h) {
  double time = -1.0;
  time = h.fcfTime();
  waitDelay("EndTestFlight", time, h);
  h.updateAFH(time);
  //h.log('t', time);
  fcfTime = fcfTime + time;
  firePropertyChange("flightTime", time);
  firePropertyChange("fcfTime", time);
  firePropertyChange("numberFCFflights", flights, ++flights);
}

/**
 *Checks to see if required number of adjustments have been made.  If so, Up
 *is called with a delay for derig time.  If not then an adjustment is made and
 *a ground check is scheduled.  Additionally, a check is made to see if anything
 *failed during this flight.  If so a call is made to the Down event.
 */
public void doEndTestFlight(Helo h){
  double derigTime = 0.0;
  double adjustTime = 0.0;
  if(!h.postFlightStatus()){//checks for failure during test
    waitDelay("CrewDelay", 2.0);
    waitDelay("Down", 0.0, h);
  }
  else if(h.getFltAdj()<=0.0){
    switch(h.imds()){
      case 1:
        firePropertyChange("OptTracker", numberAvailOptTrack, ++numberAvailOptTrack);
        break;
      case 2://non-IMDS helo
        firePropertyChange("vats", numberAvailVATS, ++numberAvailVATS);
        break;
      default: throw new IllegalArgumentException("invalid RTB gear code");
    }
    derigTime = h.derigTime();
    firePropertyChange("RTB interarrival", h.getAFH()-h.getLastRTB());
    h.setLastRTB(h.getAFH());
```

```
        h.vibesAfterRTB();
        waitDelay("UP", derigTime, h);
        waitDelay("CrewDelay", 2.0);
        firePropertyChange("rotorMaint", derigTime);
        rotorManHours = rotorManHours + derigTime;
      }
      else{
        h.setFltAdj((h.getFltAdj()-1.0));
        adjustTime = h.adjustTime();
        waitDelay("StartGroundTest", adjustTime, h);
        firePropertyChange("rotorMaint", adjustTime);
        rotorManHours = rotorManHours + adjustTime;
      }
    }

    /**
     *This method is to provide for the normal delay between the FCF crew switching
     *to a different aircraft
     */
    public void doCrewDelay(){
      //System.out.println(Schedule.getSimTime());
      firePropertyChange("fcfCrew", numberAvailFCFCrews, ++numberAvailFCFCrews);
      //System.out.println(qFCF.size()+"= FCF Q SIZE");
      if (qFCF.size()>0){
        waitDelay("StartFCF", 0.0);
      }
    }

    public void outFCF(){
      try {
          File parentDir = new File(rtb.Helo.class.getResource(".").getFile());
          File outFile = new File(parentDir, "fcf.out");
          BufferedWriter writer = new BufferedWriter(new FileWriter(outFile, true));
          writer.write(numberACFCFs+", "+flights+", "+numberRTB+"\n");
          //writer.write(card.toString());
          writer.flush();
//              FileOutputStream out = new FileOutputStream(outFile);
//              ObjectOutputStream oos = new ObjectOutputStream(out);
//              oos.writeObject(card);
//              oos.flush();
        } catch (Exception e) {
            System.out.println("Problem writing fcf.out: " + e);
        }
    }

    /**
     *String output of number of FCF crews, rotor Man hours and FCF flight time
     */
    public String toString(){
      return "Functional CheckFlight Summary:\n\tNumber of FCF Crews: "+fcfCrews
        +"\n\tNumber of VATS gear: " +vats
        +"\n\tNumber of Optical Trackers: "+optTrack
        +"\n\tTotal RTB maintenance time: "+rotorManHours
        +"\n\tFlight time spent on FCFs: "+fcfTime
        +"\n\tNumber of a/c needing an FCF: "+numberACFCFs
        +"\n\tNumber of FCF flights: "+flights
```

```
        +"\n\tNumber of RTBs: "+numberRTB;
    }


    // main method
}
```

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF REFERENCES

Akaike, H., "A New Look at the Statistical Identification Model," IEEE Transactions on Automatic Control, 19, 716-723, 1974.

Benoff, Dave, "Health and Usage Monitoring Systems," Business and Commercial Aviation, Vol 88, issue 2, New York, NY, 20 February 2001.

Buss, A. Basic *Event Graph Modeling*, Simulation News Europe, Technical Notes, April 2001.

Cioppa, T. M., "Efficient Nearly Orthogonal and Space-filling Experimental Designs for High-Dimensional Complex Models," M.S. thesis, Department of Operations Analysis, Naval Postgraduate School, 2002, Last Accessed May 2004
http://library.nps.navy.mil/uhtbin/cgisirsi/Mon+May+17+14:24:58+PDT+2004/0/520/02sep_Cioppa_PhD.pdf

Davis, E., et al., "Cost/Benefits Analysis for Integrated Mechanical Diagnostics," Report Number CSDL-R-2782, The Charles Stark Draper Laboratory, Inc, April 1997.

Duke, Alan, et.al., "The U.S. Navy/BF Goodrich Integrated Mechanical Diagnostics COSSI Program –An Update," DSTO-HUMS2001, 2001

Gupta, B.P. and Wood, E. R., " Low Vibration Design of AAH for mission Proficiency Requirements," American Helicopter Society Northeast Regional National Specialists Meeting on Helicopter Vibration, Hartford CN, November 1981.

Johnson, Lloyd, "History: Helicopter Rotor Smoothing," Last Accessed 11 June 2004, www.dssmicro.com/theory/dsrothst.htm

Law, Averill and Kelton, W. David, "Simulation Modeling and Analysis," McGraw-Hill, 2000.

Pyryt, George. "Final Technical Report for Cost/Benefit Analysis of the Integrated Mechanical Diagnostics (IMD) Health and Usage Monitoring System (HUMS)," Report 327393, The Charles Stark Draper Laboratory, Inc, 30 Nov 2001.

Rosen, A and Ben-Ari, R., "Mathematical Modeling of a Helicopter Rotor Track and Balance: Theory", Journal of Sound and Vibration, Vol. 200, No 5, pg 589-603, 1997.

Schaefer, C., and Haas, D., "A Simulation Model to Investigate the Impact of Health and Usage Monitoring Systems (HUMS) on Helicopter Operations and Maintenance," American Helicopter Society 58th Annual Forum, Montreal, Canada, June 11-13, 2002.

Veca, Angelo C., "Vibration Effects On Helicopter Reliability and Maintainability," USAAMRDL Technical Report 73-11, April 1973.

Wood, E Roberts, "An Introduction to Helicopter Dynamics." Hughes Helicopters, personally conveyed in 2003.

Naval Aviation Maintenance Plan (NAMP), OPNAVINST 4790.2H, Office of the Chief of Naval Operations, 1 June 2001, Last Accessed May 2004, http://logistics.navair.navy.mil/4790

Technical Manual, Work Unit Code, H-53E Aircraft, NAVAIR 01-230HM-8, Naval Air Systems Command, 1 August 2001.

Technical Manual, Organizational Maintenance Procedures, Vibration Manual, Navy Models, CH-53D, CH-53E, MH-53E, A1-H53CE-VIB-000, Naval Air Systems Command, 1 April 2002.

NATOPS General Flight and Operating Instructions, OPNAV Instruction 3710.7S, Office of the Chief of Naval Operations, 15 November 2001.

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center
   Ft. Belvoir, Virginia

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, California

3. Marine Corps Representative
   Naval Postgraduate School
   Monterey, California

4. Director, Training and Education, MCCDC, Code C46
   Quantico, Virginia

5. Director, Marine Corps Research Center, MCCDC, Code C40RC
   Quantico, Virginia

6. Marine Corps Tactical Systems Support Activity (Attn: Operations Officer)
   Camp Pendleton, California

7. Director, Studies and Analysis Division, MCCDC, Code C45
   Quantico, Virginia

8. Professor Arnold Buss
   Modeling, Virtual Environments and Simulation Institute
   Naval Postgraduate School Monterey, California

9. Professor Lyn Whitaker
   Department of Operations Research
   Naval Postgraduate School
   Monterey, California

10. Dr. Eric Bechhoeffer
    Goodrich Fuel and Utility Systems
    Vergennes, Vermont

11. Mr. Rich Tullos
    Naval Air Systems Command, PMA 261,
    Naval Air Station Patuxent River, Maryland

12. Professor Samuel E. Buttrey
Department of Operations Research
Naval Postgraduate School
Monterey, California